APPENDIX 3 TO
ANNEX 3 TO
AC/322(SC/1-WG/1)N(2007)0004

NATO Architecture Framework

Version 3

CHAPTER 3 NNEC Architecture Concepts and Elements

This page is left blank intentionally.

Conditions of Release

With reference to NATO Documents C-M(2002)49 and AC/322-D/1, this document is released to all parties concerned at the direction of the NATO Consultation, Command and Control Board (NC3B) subject to the following conditions:

1. The recipient party agrees to use its best endeavours to ensure that the information herein disclosed, whether or not it bears a security classification, is not dealt with in any manner (a) contrary to the intent of the provisions of the Charter of the NATO C3 Organisation, or (b) prejudicial to the rights of the owner thereof to obtain patent, copyright or other likely statutory protection thereof.

2. If the technical information was originally released to NATO by a NATO or Partner Government subject to restrictions clearly marked on this document, the recipient party agrees to abide by the terms of the restrictions so imposed by the releasing Government.

3. This material may be reproduced free of charge in any format or medium provided it is reproduced accurately and not used in a misleading context. Where this material is being republished or copied to others, the source of the material must be identified and the copyright status acknowledged. For further information, contact NATO at http://www.nato.int.

This page is left blank intentionally.

Reader's Guide

Background

An architecture is the fundamental organisation of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution. An Architecture can be captured in a formal description of an instance, or configuration of people, processes, systems and organisations, connected by their inter-relationships. An architecture description includes views showing various aspects of the architecture. The views include architectural elements and the relations between elements as governed by a Metamodel.

Architectures can be represented by models. These support operational processes by providing an explicit representation of the operational domain that can be used in analysis, for articulation of issues and requirements, as support to planning, and as a means of solution design and validation, amongst other things. Architectures can be developed for the smallest subsystem up to and culminating in an architecture that covers the whole enterprise.

The role of an enterprise architecture is to provide decision support, in the context of the enterprise strategy, for the use of resources (including processes and procedures) in the enterprise. In other words, the architecture is responsible for defining how resources will be used to support enterprise strategy and benefit the NATO goals and objectives. Enterprise architecture touches every part of the organisation.

Architectures are to be used as analysis tools to develop new capabilities, structure organisations and to optimize processes and spending. There is an increase in the need for international coalition operations (NATO Response Force) and a growing need to deliver end-to-end capability, whilst delivering more for less and ensuring interoperability. NATO Network Enabled Capability (NNEC) is a key part of meeting this changing need, and enables us to federate systems, sensors, effectors and hence improve military effectiveness.

There is a need for a more structured approach to manage the complexity whilst balancing all appropriate user perspectives. Architectural frameworks support this goal, and the most mature and widely adopted architectural frameworks in the defence sector are the US DoD Architecture Framework (DoDAF), and the UK MOD Architecture Framework (MODAF). NATO is using its own NATO Architecture Framework (NAF) to support this goal. The previous version of NAF was tightly coupled to DoDAF. The current version has been updated in several areas, and has incorporated not only US and UK experiences, but also views and experiences from other Nations, from industry and from academia.

New features include:

- initial support for NNEC architectures
- support to the NATO capability development process;
- support to programme management;
- integration of Service-Oriented Architecture (SOA) concepts;
- support to spectrum and bandwidth planning;
- provision and integration of the NATO Architecture Framework Metamodel (NMM) and the NATO Architecture Repository (NAR);
- a running example.

The NAF is not an architecture itself, but it provides the rules, guidance, and product descriptions for developing, presenting and communicating architectures. The NAF is also the common denominator for understanding, comparing, and integrating architectures. The application of the framework will enable architectures to contribute most effectively to the acquisition and fielding of cost-effective and interoperable military capabilities. The framework ensures that the architectures developed by NATO and the Nations can be compared and related across NATO and National boundaries. NATO common funded programmes are to comply with the NAF to promote systems interoperability.

NAF Document Suite Structure

The NATO Architecture Framework is composed of ten volumes, covering NAF-chapters 1 through 7, and NAF-annexes A through C (see Figure 3-1). Each NAF-chapter and each NAF-annex is contained within a separate volume, and each volume is published as one electronic file. Each electronic file is intended to be readable as a stand-alone document. For that purpose, each volume contains common introductory sections that provide enough context to understand the contents of the NAF as a whole, as well as that particular volume. This approach enables us to disseminate concise portions of information to the intended audience. The NAF will also be published in HTML format.

Executive Summary
In addition to the ten volumes, an executive summary exists. This is a concise summary of NAF version 3. The document is intended to give the reader a quick — but nonetheless comprehensive — understanding of NAF, its use and benefits. The summary is sufficient to understand why the NAF is relevant and where and how the NAF can be used. The details can then be found in the NAF-volumes themselves.

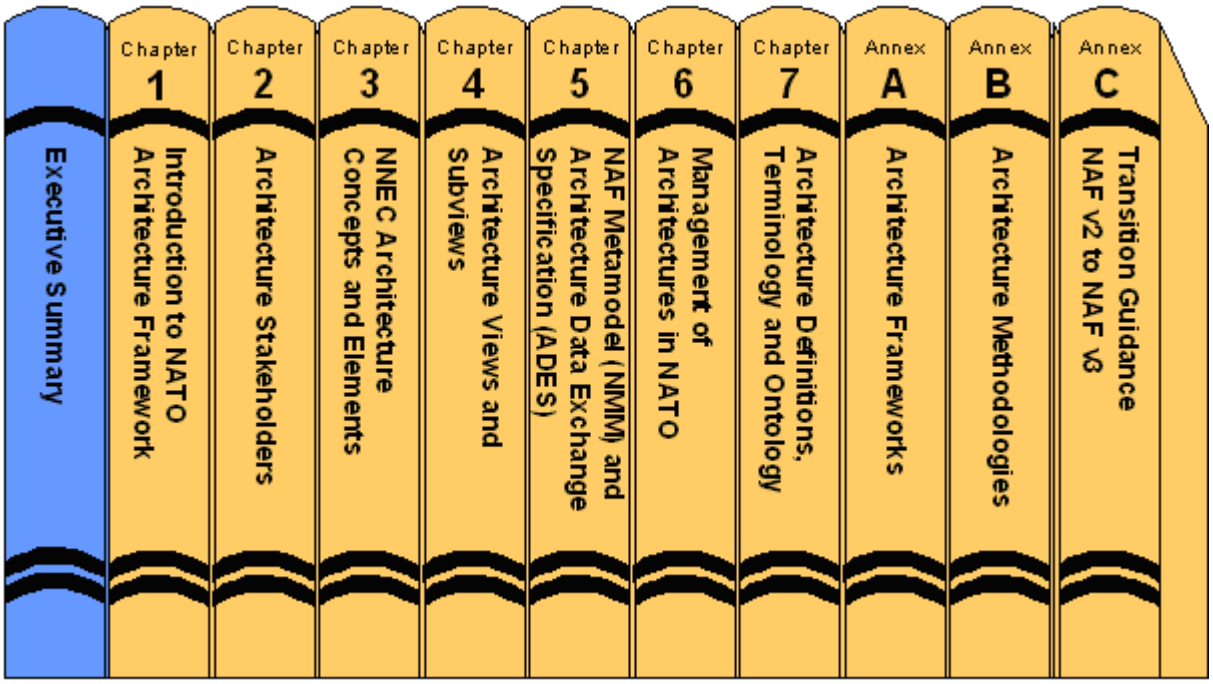

Figure 3-1, NAF v3 Document Suite Structure

CHAPTER 1 Introduction to NATO Architecture Framework

NAF-chapter 1 is the introduction to the NAF. This chapter essentially presents the business case for NATO architectures. Chapter 1 defines what an architecture is, what the benefits are, and why architectures are relevant to NATO in the light of current and future developments, such as NNEC, NRF and Comprehensive Approach. This chapter contains the following paragraphs:

1.1 Introduction to Chapter 1

1.2 What is an Enterprise Architecture?

1.3 Purpose and Scope of the NAF

1.4 Imperative Documents

1.5 Why use NAF?

1.6 The NATO Architecture Framework (NAF)

1.7 New Features and Important Changes in NAF

1.8 Architecture Tools

1.9 Types of NATO Architectures

1.10 NATO Architecture Views

1.11 Management of NAF and architectures

1.12 Guidelines for Architects

CHAPTER 2 Architecture Stakeholders
NAF-chapter 2 addresses the stakeholders and communities of interest that may benefit from the use of the NAF. It is one of the principles of this version of the NAF to align the architecture framework with the objectives and interests of those who use it, providing for as much added value as possible, without being sidetracked by issues that do not directly contribute to the stakeholder's cause. This chapter contains the following paragraphs:

2.1 Introduction to Chapter 2

2.2 Identification and Description of Stakeholders and Communities of Interest (CoIs)

2.3 Requirements Analysis of Communities of Interest (CoIs)

CHAPTER 3 NNEC Architecture Concepts and Elements
NAF-chapter 3 provides a description of the NNEC architecture concepts and elements that are essential for developing a complete and coherent architecture, capable of specifying seamlessly interoperable network enabled systems. This chapter contains the following paragraphs:

3.1 Introduction to Chapter 3

3.2 NNEC Architecture Concepts

3.3 NNEC Architecture Elements and Descriptions

CHAPTER 4 Architecture Views and Subviews
Chapter 4 defines a toolbox of architecture views and subviews, where each subview comprises of specific diagrams and specifications, intended to support a specific purpose, and intended to be communicated to specific stakeholders and specific Communities of Interest. This chapter contains the following paragraphs:

4.1 Introduction to Chapter 4

4.2 NATO Architecture Views

4.3 NAV, NATO All View

4.4 NCV, NATO Capability View

4.5 NOV, NATO Operational View

4.6 NSOV, NATO Service-Oriented View

4.7 NSV, NATO Systems View

4.8 NTV, NATO Technical View

4.9 NPV, NATO Programme View

4.A Mapping Subviews to Communities of Interest

4.B Running Example

4.C Mapping Subviews to NNEC Elements

CHAPTER 5 NATO Architecture Framework Metamodel (NMM) and Architecture Data Exchange Specification (ADES)

NAF-chapter 5 contains the NAF Metamodel (NMM). In essence, an architecture is delivered as a set of design models, specifications and guidelines, which are indispensable for subsequent system development within NATO's highly complex, dispersed and heterogeneous automated information system. In practice, the result of an architecture effort is an architecture core data repository. The NAF Metamodel is the engine of this repository. The NAF Metamodel is also intended to facilitate exchange of architecture design information. This chapter contains the following paragraphs:

5.1 Introduction to Chapter 5

5.2 NATO Architecture Framework Metamodel (NMM)

5.3 NATO Architecture Data Exchange Specification (ADES)

CHAPTER 6 Management of Architectures in NATO

NAF-chapter 6 provides guidelines on how to manage NATO-architectures. Merely developing an architecture does not provide or ensure the full benefits. Architectures must also be maintained, communicated, and enforced. These aspects of the architecture life-cycle are the subject of NAF-chapter 6. This chapter contains the following paragraphs:

6.1 Introduction to Chapter 6

6.2 The Management and Use of NAF v3

6.A General Guidelines for the Management of NAF v3 Architectures in Nations

6.B NAF v3 Request For Change Proposal (RFCP) Procedure

6.C Standardization of the NATO Architecture Framework Metamodel (NMM) and the Architecture Information Exchange Mechanism (ADEM)

CHAPTER 7 Architecture Definitions, Terminology and Ontology

NAF-chapter 7 contains a comprehensive glossary of terms. All essential terms used in the NAF are defined in a precise and unambiguous way to optimize consistency, and increase readability of each NAF-volume, and the entire framework, across the different volumes. This chapter contains the following paragraphs:

7.1 Introduction to Chapter 7

7.2 Definitions

7.3 Acronyms and Abbreviations

7.4 Taxonomy (not yet included in NAF version 3)

7.5 Ontology (not yet included in NAF version 3)

ANNEX A Architecture Frameworks

NAF-annex A presents an overview of some architecture frameworks. This chapter presents brief introductions to other architecture frameworks for ease of reference and for comparison. This chapter contains the following paragraphs:

A.1 Introduction to Annex A

A.2 The Open Group Architecture Framework (TOGAF)

A.3 Gartner's Enterprise Architecture Framework

A.4 Zachman Framework

A.5 Department of Defense Architecture Framework (DoDAF)

A.6 Ministry of Defence Architecture Framework (MODAF)

A.7 AGATE v3

A.8 Defence Architecture Framework (DAF)

A.9 Model-based Architecture for Command and Control Information Systems (MACCIS)

ANNEX B Architecture Methodologies

NAF-annex B contains an overview of some architecture methodologies. This chapter presents brief introductions to other architecture methodologies for ease of reference and for comparison. NC3A's Architecture Engineering Methodology (AEM) is included in full. This particular methodology serves as information for those Partner- and NATO-Nations that want to know more about NATO architecture projects where NC3A is host Nation or provider, and where the AEM is used. This chapter contains the following paragraphs:

B.1 Introduction to Annex B

B.2 TOGAF ADM

B.3 Boeing/Openwings

B.4 META Group and Gartner process model

B.5 DoDAF's 6 step model

B.6 NC3A/AEM

ANNEX C Transition Guidance NAF v2 to NAF v3

NAF-annex C provides for guidance on how to migrate existing architectures, based upon NAF version 2, to architectures that are compliant with NAF version 3. This chapter contains the following paragraphs:

C.1 Introduction to Annex C

C.2 Considering Transition Feasibility

Table of Contents

3.1 Introduction to CHAPTER 3

3.1.1 Objectives

The objective of this chapter is to provide guidance and support for developing underpinning NNEC architectural elements that can be captured and stored in a knowledge base and can be reflected according to the views and subviews as described in Chapter 4.

Although the NAF does not prescribe the methodology that needs to be followed to arrive at sound, rigorous and well-reasoned deliverables, it wants to provide support for the production of deliverables that are coherent, consistent and relevant. This chapter outlines a series of NNEC architecture elements, taken from best-practices in the field of architecture in industry and academia, which aim at capturing and structuring the essential aspects of the universe of discourse of the architecture effort at hand.

This chapter describes the essential NNEC architecture elements and their interrelationships. The benefit of identifying these elements is that it allows for:

- Capturing those elements that are essential for any NATO architecture, especially the ones that support the NNEC development.
- Capturing essential NNEC architecture elements independent of their representation for various stakeholders.
- Segmentation of complexity into manageable but focused descriptions.
- Traceability from operational requirements to technology implementation.
- A sound reasoning about the NNEC architecture elements and their interdependencies underpinning the representation for the various stakeholders.
- Linking architecture development to other NATO processes like Operational Planning and Capability Management

The views and subviews described in Chapter 4 are expected to be the deliverables that many stakeholders are most interested in. It is essential that the views and subviews are produced based on a sound rationale in order to have consistent and coherent diagrams and descriptions.

This chapter provides a starting-point by presenting a series of NNEC architecture elements and descriptions that allow a strong and traceable line of reasoning from understanding the mission space to the required technology. The artefacts produced can be used in different representations and different combinations to produce consistent and coherent views and subviews.

It is important to use the NNEC architecture elements of this chapter as the starting point.

All architecture elements listed in this chapter have been proven to be useful in the transformation towards Capability-based planning and NNEC.

3.1.2 Scope

The scope of this chapter is to list a comprehensive, consistent and coherent series of descriptions of NNEC architecture elements as required for architecting C3 systems support. The series of descriptions is suitable for capturing the operational aspects essential to determine the requirements for the C3 systems support and translate these in the contents and structure of C3 systems. The contents and structure of the C3 systems are reflected in two additional aspects: the information aspect covering the information required by the users and the technology aspect containing the systems' functionality and corresponding technical facilities to create, process, store and distribute the information.

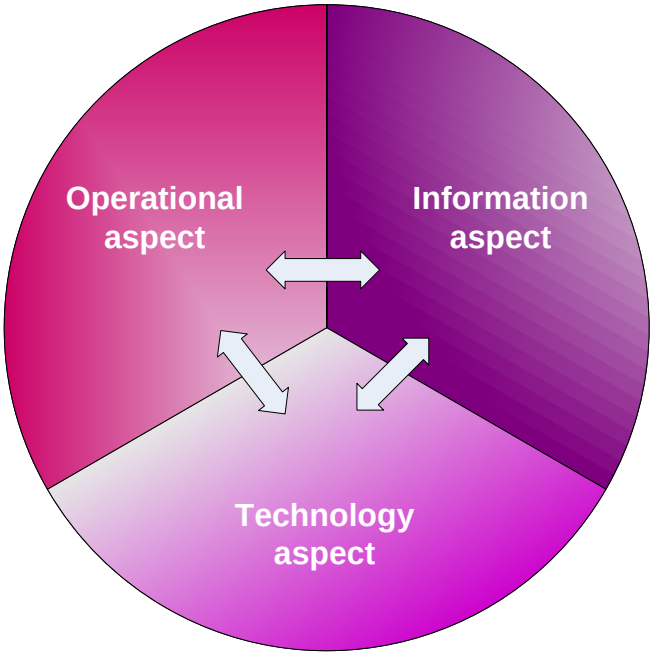

Figure 3-2, Scope architecture elements

Operational aspect

In order to develop NNEC architectures it is crucial to have a sound understanding of the operational domain, its structure and behaviour. In the first place one should understand the goals and objectives of the domain. Furthermore, the processes and products that are to be delivered to the environment must be understood. For that it is also relevant to have insight in the capabilities that are required for an adequate

delivery of the products. Finally, the business rules that constrain the delivery must be taken into account.

Information aspect

The objective of capturing elements of the information aspect is to understand what information is required by whom, how the information is used and created. The starting point for the elements of the information aspect is the knowledge and insight of the operational domain.

Technology aspect

The operations and the required information will be supported by applications. In the technology aspect architecture elements are captured that describe and structure the required services, applications and the underlying infrastructure.

3.1.3 Principles, Tenets and Assumptions

The NNEC architecture elements and descriptions in this chapter are founded on a set of tenets and principles. Each of those principles will be briefly described.

Embedding Architecture

Every architecture description is developed for a specific purpose. In fact, there needs to be a well-defined set of questions that an architecture description needs to provide the solution to. Furthermore, the architecture effort can be guided and constraint by specific requirements. These requirements are of a different nature then the considered system. E.g. there can be a constraint that requires the architecture description to define an automated system for tracking goods during deployment to theatre that can be implemented within 2 years. The time-constraint does not originate from logistics, rather from programme management.

Although these questions are essential for an architecture effort, they are not part of the architecture. They provide the linkage between a strategy for a particular system (used in the narrower sense as automated system or in the broader sense as the operational system) and the architecture that carries the solutions for implementing that strategy.

Therefore, it is important to record these questions and constraints with the architecture description.

The resulting architecture description needs to be put to use to transform (parts of) the organisation. Consequently, there needs to be a linkage from the architecture effort to programme management, long-term and operation planning activities and decision-makers.

These linkages to strategy and application of architecture are not explicitly addressed in this chapter.

NNEC Service Framework

The areas to be covered by the NNEC Architecture work were identified during the conduct of the NNEC Feasibility Study. These areas are illustrated in the following diagram.

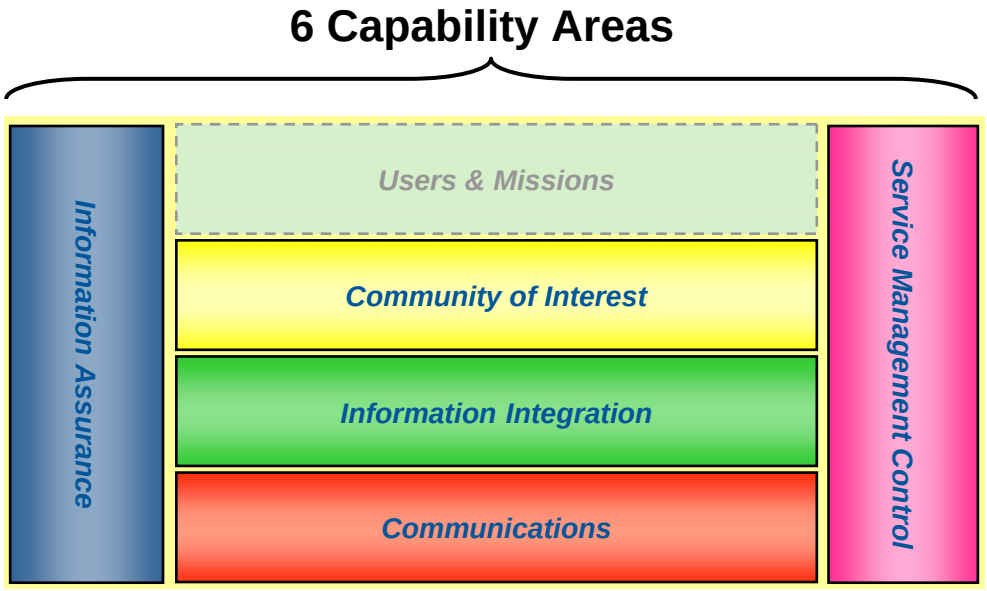

Figure 3-3, NNEC capability areas

Six areas, referred to as Capability Areas, were identified. Four of the areas, Communications, Information and Integration, Information Assurance, and Service Management and Control, are linked to the NII (Networking and Information Infrastructure. The Communities of Interest Area relates to traditional communities, such as Land, Air, Maritime, Joint C2, Intel, and Logistics, as well as new, or ad-hoc communities of interest that may form between Nations in response to mission needs. The final area, Users and Missions is linked to evolutionary sets of objectives expressed in terms of operational capabilities and associated concepts of operations.

The Capability Areas reflect the overall structure of capabilities and their supporting services at a highest level of abstraction. In essence the areas are independent and there is no relationship between them. An architecture description must reflect the line of reasoning how to arrive at the contents of these Capability Areas. In essence and leaving out all of the more intricate details, this line of reasoning uses four steps to arrive at the desired services and their internal construction:

- Understand what NATO's mission space is all about and construct how to operate and with what to implement the operational objectives;

- Derive what information is required for executing the operations and how to construct the best way of distributing and managing the information and with what to implement this;
- Derive the required automated support for handling the information and construct the best way to run and manage and implement the automated services;
- Derive the required common and generic facilities for executing the automated services and construct the best way to implement and manage this infrastructure.

During these four main steps Information Assurance and Service Management and Control is constantly considered and addressed as integrated parts of the solutions.

The line of reasoning outlined above is best suited for the top-down reasoning as is relevant for the architecture core data at the overarching architecture level of description. However this reasoning can also be applied in reverse, as is more applicable to the Reference Architecture level of descriptions:

- Understand which facilities are planned or already implemented and how they are constructed (e.g. how common and generic are they);
- Derive the automated support that can be offered to the users and where are the limitations in functionality and information handling that need resolution and construct the best way to run and manage and implement the automated services;
- Derive the information for execution the operations that can be handled by these automated services and construct the best way of distributing and managing the information and with what to implement this;
- Derive to which extend NATO's operations and objectives can be supported by this information handling.

3.1.4 Approach

In the remainder of this chapter, the architectural elements that are considered to be essential for an NNEC architecture are described.

For each architecture element, the following structure will be applied:

- Rationale; depicting the underlying ideas of the architecture element.
- Definition; what is the meaning of the architecture element?
- Objective; what is the aim for capturing the architecture element?
- Description; providing an explanation what the architecture element is.
- Added value; for what other purposes is the architecture element useful.
- Similar terms; what names are used in different approaches. Of course different definitions and descriptions are used by the different approaches for similar

architecture elements. The following approaches have been consulted: DoDAF, MODAF, DAF, AEM, Openwings (Boeing), IBM, Gartner, Zachman, TOGAF.

- Covered aspects; to what architecture aspects the architecture element contributes to. The architecture aspects are operational, information and/or technology aspect.
- Corresponding architecture description levels; to what architecture description levels the architecture element belongs to. The architecture description levels are functionality level, construction level and implementation level.
- Example; providing a descriptive example of the architecture element. The examples shown are depicting only a diagram of the corresponding NNEC architecture element. These examples are not examples of (sub)views.

Capturing and describing architecture elements can be done through several types of architecture artefacts, like diagrams, tables, matrices and plain text descriptions. For each of the architecture elements an example is presented.

Although best guidance to architecture development is given by a methodology the NAF does not prescribe any. The descriptions in this chapter are the closest guidance the NAF can provide to the architecture development process.

3.2 NNEC Architecture Concepts

In order to develop architectures that support the NNEC transformation four concepts has been used to define and describe the relevant NNEC architecture elements. These concepts are:

- Service-oriented development
- Component-based development
- Architecture description levels
- Quality factors

3.2.1 Service-Oriented Development

The Service-Oriented Development (SOD) paradigm has gained a lot of momentum and is identified by the NNEC Feasibility Study as an important paradigm for the transformation towards NNEC. Although very well applicable to systems development, the principle of customers, suppliers and freedom of choice is originally found in the day to day 'business' environment. The descriptions of the architecture elements in this chapter include this paradigm in each of the three aspects, i.e. operational, information and technology aspect. This allows for consistency and coherence between the different descriptions and aspects. In order to indicate the generic use of the paradigm we use the term Service-Oriented Development (SOD).

3.2.2 Component-Based Development

The service-orientation paradigm was preceded by the concept of components and the component-based approach. This trend is still strong today. Systems are designed as pieces collaborating to form an integrated whole, in order to accommodate distribution of software across heterogeneous, dispersed platforms; to integrate functionality from many sources, including legacy systems; to spread development across numerous multi-disciplined teams, and parallelize development across continents; and to reduce complexity and mitigate risks. Just to name some of the benefits.

Because of these benefits, service-oriented designs, large-scale enterprise-wide solutions, and web-based applications, must be component-based. The main challenge of the component-based approach can be found in the concern on how to carve up functionality and responsibilities into pieces of software that can be designed and built separately; that can be packaged and distributed, and that can be provisioned with the right interfaces and mechanisms to enable correct collaboration with other pieces of software.

Crucial to designing component-based systems is the distinguishing of an implementation-independent specification of components, which allows the design of a solution that has not been impeded by implementation constraints. As long as other components access a component through its specified interface, the developer is free to choose any implementation, as long as that implementation satisfies what the interface promises, both in functionality, and in level of quality (e.g. performance, and other quality requirement categories). By implication, it is also crucial to design an infrastructure that is capable of running and maintaining a well-defined component model.

Component-based design focuses mainly on three aspects:

- On specifications, where logical components collaborate to supply a solution that will provide the right functionality at a right level of quality.
- On implementation, where feasibility of implementation is considered, and where the design solution will be populated with bespoke and of-the-shelf products that may deploy multiple different technologies.
- On distribution, where many variations on deployment are possible, and that will greatly influence the implemented solution's performance.

Especially in an NNEC environment, where it is necessary to design and build systems in a modular and evolutionary way, it is crucial to incorporate the component-based approach in all aspects of design and development. Subsequently, the component-based approach has been incorporated in the NAF.

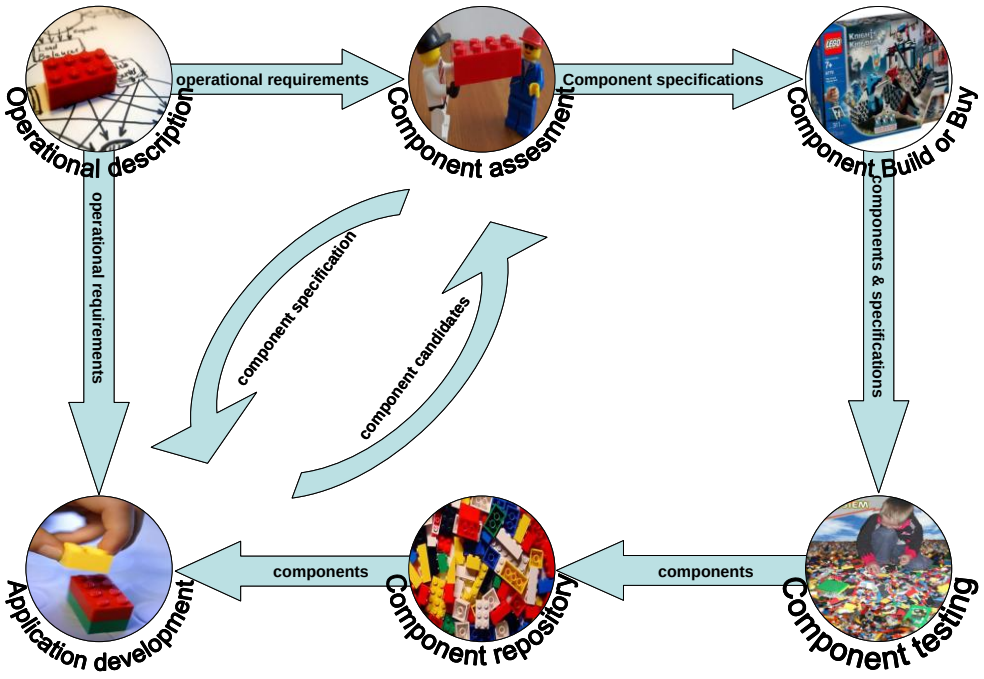

Figure 3-4, Component-based development

3.2.3 Architecture Description Levels

In an architecture many architecture elements need to be identified, defined and described. In order to structure the architecture elements, three architecture description levels are helpful:

- Functionality description level
- Construction description level
- Implementation description level

Functionality description level

The functionality description level handles all NNEC architecture elements that capture the required functionality of the considered system or aspect system. E.g. it contains descriptions that capture what people need to be able to do in the mission space (without stating how they need to do those things) and it also contains descriptions that capture what the applications needs to be able to do in order to support the people in the mission space. The functionality description level acts as the storage for requirements for each of the considered systems and aspect systems.

Construction description level

The construction description level handles all NNEC architecture elements that capture how the required functionality, described in the previous description level, ideally should be constructed. These elements are of a more compound nature, since they build on the results of the elements in the functionality description level. The constructions captures at this layer are independent of their actual implementation. As a sound design principle, the solutions envisioned here are not constrained by imperfect or unavailable technology. These solutions act as the ultimate goal to strive for. The solutions do refer types of technology required for the construction, but do not refer to actual technology products.

Implementation description level

The implementation description level handles all NNEC architecture elements that capture with what the solution is actually implemented. The elements at this level reflect the feasibility of the solution. At this level actual technology products are considered and the consequences of their typical behaviour are taken into account. This usually implies a reduction or limitation to the envisioned ideal solution. This could be a temporary situation, e.g. R&D is in progress to identify the better technology, while in the meantime other, less capable technology is implemented.

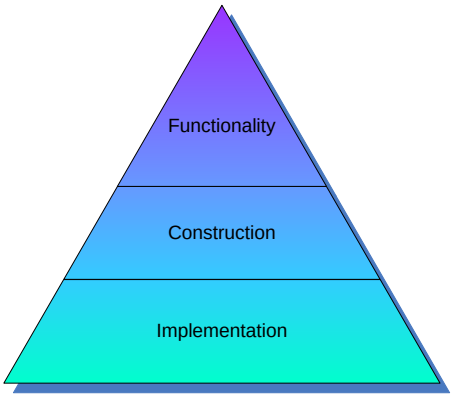

Figure 3-5, Architecture description levels

3.2.4 Quality Factors

For all operations, in the real world and in automated systems, there are conditions that apply on top of the actual operation that is conducted. The fact that a task needs to be accomplished within a certain time limit possesses additional requirements to the support of that task. Those conditions are also referred to as 'non-functional requirements'. These conditions are important to an architecture because they impose additional functionality. E.g. the fact that information can only be accessed by a person that has the appropriate access rights implies functionality that checks these rights. In fact, honouring these conditions determine the usability and quality of the support offered by the C3 System. Therefore, these conditions are to be captured as Quality Factors that need to be fulfilled.

By identifying the Quality Factors in the operational domain and translating them (using the Representation paradigm) through the aspect systems down to the implementation of the technology, coherence, consistency and traceability can be reached in the design decisions made in the architecture.

3.3 NNEC Architecture Elements and Descriptions

In order to describe an NNEC architecture correctly and sufficiently the following architecture elements need to be captured:

- Actor: who are the parties responsible for executing tasks?
- Operational objective: what are the essential goals that need to be reached?
- Capability: what abilities are required for conducting operations in order to reach desired effects?
- Operational object: what entities are needed to reach the objectives?
- Information object: what facts about the operational objects are relevant?
- Process: what operational processes are needed and how are they orchestrated in order to deliver a desired an end-state?
- Location: where do all the processes, services and activities take place?
- Information requirement: what information is needed?
- Business rule: what are the conditions and constraints while delivering the end-state?
- Information product: what are the information deliverables?
- Information flow: What is the end-to-end chain of information between the parties?
- User: who are the parties that require functionality of applications?
- Service: what is the delivery of (information) products and functionality?
- Component: What is the construction of services?
- Component collaboration: How do the components interact?

3.3.1 Actor

Rationale

In order to be able to allocate responsibilities in a coherent and non-conflicting manner to actual persons, the responsibilities need to be captured first, irrespective of the actual persons or executing parties. The complete set of these responsibilities needs to cover all of NATO's external and internal obligations. In order to fulfil his obligations, an actor has a certain information need. By linking these information requirements to the responsibility instead of to an individual, flexibility and applicability of information requirements become independent of an incidental

organisational structure. The structure (e.g. for a new mission) can change, but as soon as the responsibilities are allocated to individuals at certain locations, the information requirements are known. Due to the linkage of these information requirements to responsibilities, the conditions (e.g. timeliness of information, availability, etc.) for fulfilling the requirements also become clear and stable.

Definition

An actor is an implementation independent unit of responsibility that performs an action to achieve an effect that contributes to a desired end state.

Objective

The objective of capturing the actors is to gain a thorough understanding of the distribution of responsibilities as a basis for identifying the collaborations in the organisation.

In order to arrive at interoperable parts of a larger system it is essential to start with an implementation independent description. At a later stage of the architecture development, that will be the neutral point of reference to measure and identify the interoperability requirements.

An actor is independent of the way how the responsibilities are allocated to people or organisational units, thus establishing a single point of reference for the operational responsibilities, while allowing freedom and flexibility for different implementations.

Description

Within the operational domain, a set of independent entities are responsible for performing services in order to reach operational objectives. These parties are called actors. Each actor can perform one or more roles. An actor can be a consumer of a service. This is typically a role of a commander. An actor can also be the producer of a service. This is typically the role of those that are ordered to conduct a task by a commander. An actor can perform several roles. E.g. a commander can require a service from his subordinates but he is also delivering a result to his higher echelon commander.

When implementing an actor, one actor role can be assigned to one or more individuals. And one individual can be assigned one or more actor roles. These assignments needs to be carefully considered since there can be a conflict of interest between the responsibilities.

Identifying the different roles and responsibilities is essential for further architecture modelling of the information requirements that pertain to them.

Added value

The added value of capturing Actors is that it allows:

- To delimit the scope of the operational contents of the mission space.
- To understand responsibilities internal to the system and those in its environment.
- To have a better understanding of the internal dependencies between parties.
- To act as an entry-point to identify information requirements.
- To create a basis for agreements between the Nations and other contributors (third parties) in order to identify responsibilities.
- To help assign responsibilities and authorities to actual persons and organisational entities.
- To serve as a stable starting point for process and organisation design.

Similar terms

For the concept of the architecture element ‘Actor’ different names are used. Common similar terms are:

- Business actor
- Performer
- Logical Node

Covered aspects

The architecture element ‘actor’ is part of the operational aspect, since it refers to elements of the real world. Instances of an actor are e.g. individuals or a group of individuals.

Corresponding architecture description level

An actor is an implementation independent concept, depicting what the responsibility is. Therefore the architecture element ‘actor’ consists of the functional level.

Example

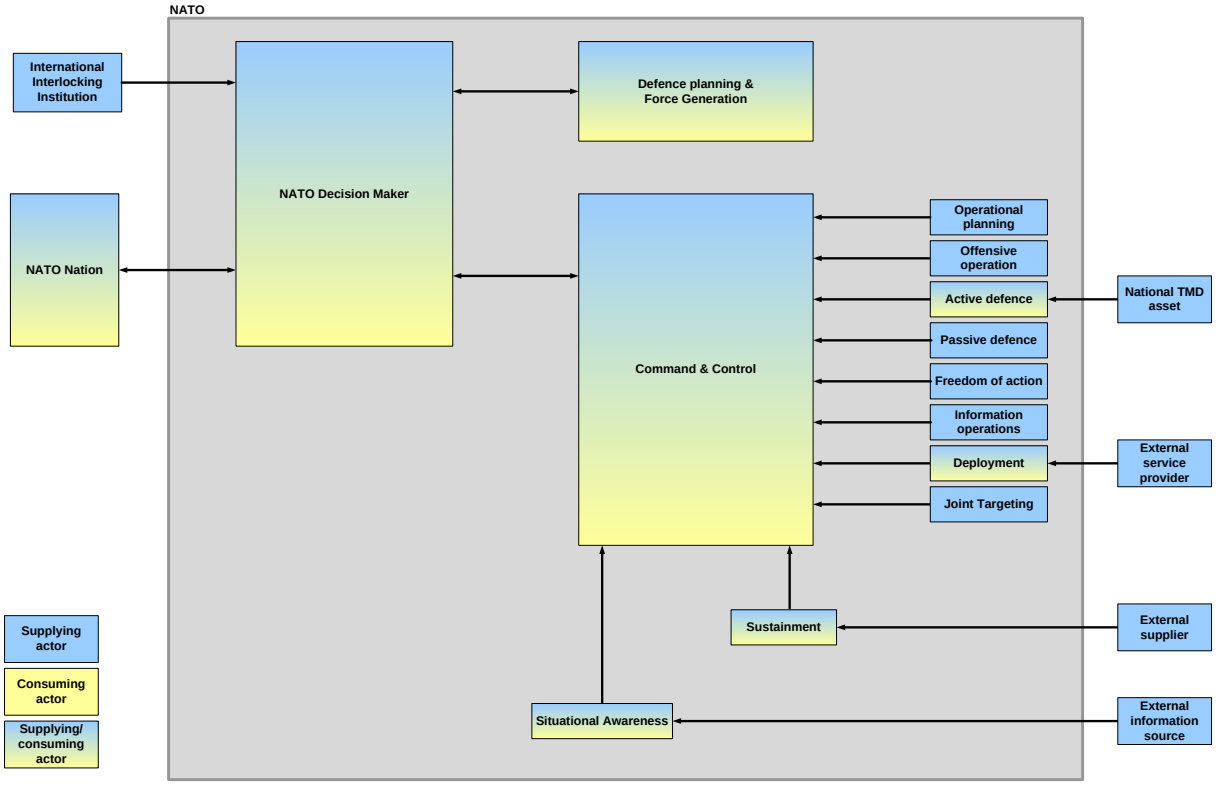

Figure 3-6, Actors from Overarching Architecture v2.5

3.3.2 Operational Objective

Rationale

Anything actors do needs to contribute to the goals of the organisation. This provides a means to validate whether all responsibilities and tasks of the actors indeed contribute to at least one goal of the organisation, and whether all goals are taken care of by the responsibilities and corresponding tasks of actors. The operational objectives are enduring, they do not change easily over time, and e.g. the ultimate goal of NATO has not changed since the alliance was established[1]. Since actors, responsibilities and the tasks they conduct are founded on the goals, this provides for a good stable foundation for an architecture.

[1] NATO's essential purpose is to safeguard the freedom and security of all its members by political and military means in accordance with the North Atlantic Treaty and the principles of the United Nations Charter. (NATO Handbook).

Definition

An operational objective is the intent of authority to reach a certain end state.

Objective

The objective of capturing the operational objectives of the domain is to establish an enduring stable basis for identifying and defining processes and capabilities, including information needs, automated systems support, organisational structure, etc.

Description

Operational objectives are closely linked to effects. An organisation can only be effective in case all tasks and processes contribute to the operational objectives, and therefore contribute to desired effects.

Operational objectives are enduring overtime and are not dependent on how the organisation is structured and implemented. In other words, operational objectives give guidance for structuring and implementing matters such as required capabilities, task descriptions, processes, organisation structure.

Understanding the operational objectives allows for the validation of tasks and responsibilities in that organisation. It should be possible to allocate all processes, tasks and responsibilities to the operational objectives.

Added value

The added value of capturing operational objectives is that it allows:

- To delimit the scope of operational contents of the mission space by identifying all the objectives in the domain.
- The collection of operational objectives serves as an anchor point for adaptability to change in implementation dependent parts of the organisation. Therefore, it is highly beneficial to base architectural designs on operational objectives.
- To establish a coherent structure of effects, both in peacetime and in wartime
- To test a domain on its effectiveness, e.g. ‘what processes are contributing to this operational objective?’, ‘where has the responsibility for this operational objective actually been assigned in our organisation?’, ‘which automated information systems are (to what extent) contributing to this objective?’ This in turn will detect overlap in organisation, processes and IT.
- To give direction to the content of the performance indicators in order to realize a desired level of execution.

Similar terms

For the architecture element ‘operational objective’ the following similar terms are used:

- business goal
- business objective
- operational goal

Covered aspects

Obviously, the element ‘operational objective’ is part of the operational aspect.

Corresponding architecture description level

As previous stated, an operational objective is an implementation independent concept, describing the enduring goals of an organisation and therefore contributes to the description of the functional level.

Example

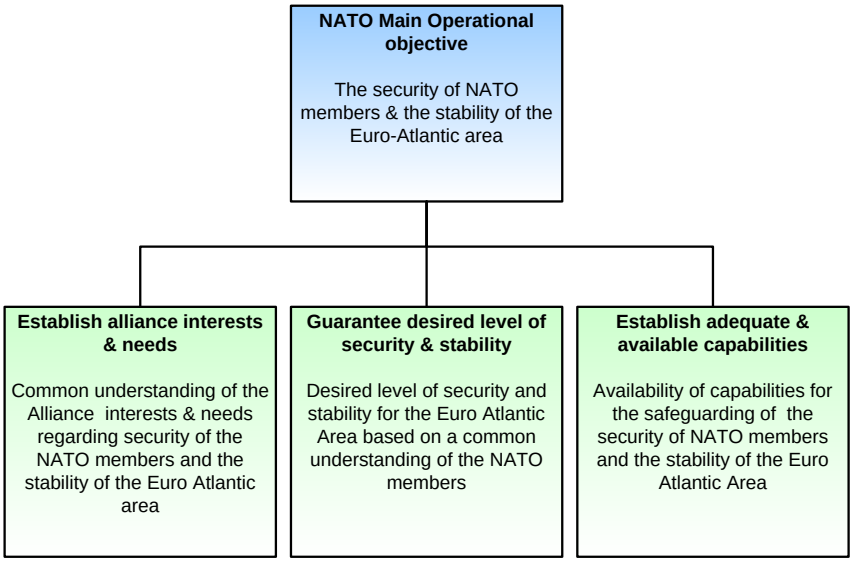

Figure 3-7, NATO's main operational objectives

3.3.3 Capability

Rationale

For achieving the goals and objectives, people need capabilities with certain functionality and implementation into resources. By understanding the way the goals and objectives are reached next to the quality of operations requirements, capabilities can be identified. In fact, the required capabilities can be optimised to support the achievement of the goals and objectives effectively and efficiently.

Definition

Capability is defined as the ability to deliver a specified type of effect or a specified course of action.

Objective

The objective of identifying ‘capabilities’ is to understand what abilities the resources (people and assets) need in order to deliver the required effects.

Since Capability Based Planning (CBP) is crucial for the implementation of the Strategic Vision, capability management is of great importance. Currently, as part of capability management, long-term and mid-term plans are being developed.

Description

Delivering a specified effect requires one or more capabilities. This means that the specified effects lead to the identification of capabilities. Taking the scope for the architecture elements into account, two types of capabilities can be distinguished: operational capabilities that refer to the execution of tasks and processes, and system capabilities regarding the required functionality and technical infrastructure.

An NNEC Architecture is developed a/o to identify the required C3 support to operations. The identification, definition and development of capabilities can benefit highly from an NNEC architecture but requires additional expertise and should be conducted as a collaboration with the architects and defence and capability planners, in order to cover the remaining aspects of the DOTMLPFI spectrum.

By analysing the responsibilities of all the different personnel (military and civilian, operational and supporting) the NATO objectives they contribute to can be established. If conducted at the level of every individual in NATO this would be an enormous task. Therefore in an NNEC Architecture abstract roles needs to be identified. Although these roles represent actual people, they are described such that they are generic and independent of the number of persons that execute each role

and independent of the person(s) that execute these roles today. For this person-independent nature the term ‘actor’ is used.

Each actor can be linked to what he needs to accomplish, in terms of tasks with matching tangible results and effects, and the objectives these results and effects need to contribute to. In fact the reasoning is much more top-down then it might seem. First NATO objectives have been captured and then the required results and effects have been identified and subsequently the required actors. This triad of objectives-results-actors leads to the identification of required capabilities. By following the strand from objectives through results to actors with specific capabilities to actual individuals, the initial question ‘who needs which capabilities?’ can be answered supported by a clear and rigorous reasoning.

This line of reasoning can be applied equally for the identification of system capabilities. In order to make optimal use of its capabilities, the systems that support the actor, requires specific capabilities regarding the system functionality and technical infrastructure.

Added value

The added value of capturing ‘capabilities’ is that it allows:

- To delimit the scope of the contents of both the operations and the C3 systems.
- To have a better understanding of the interdependencies between parties and C3 systems of both NATO and Nations.
- To create a basis for agreements between Nations and other contributors (third parties) in order to enable the sourcing discussion ‘what does NATO need to do by itself and what could/should other parties do?’
- To support the defence and force planning process.
- To help assign tasks to actual persons and organisational units.

Similar terms

For the concept of ‘the ability to deliver a specified type of effect’ the most common approaches for defence all use the term ‘capability’.

However, for the different types of capabilities different names are used. For ‘operational capability’ the term ‘military capability’ is used and ‘equipment capability’ is used for ‘C3 system capability’.

Covered aspects

The architecture element ‘capability’ is covers two architecture aspects, since it refers to elements of operation (capabilities of actors) and elements of technology (C3 system capabilities).

Corresponding architecture description level

Capabilities refer to properties of an actor or a system in order to deliver the required effects. In that sense ‘capability’ is an element that is part of composite parts that constitute an actor or a C3 system. And therefore, the concept of ‘capability’ is part of the construction level.

Example

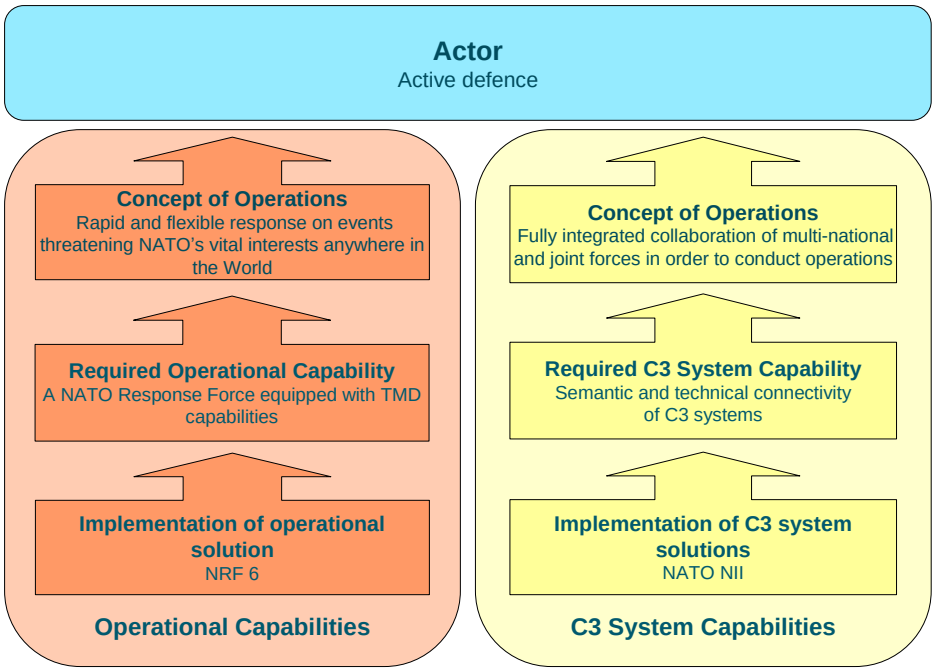

Figure 3-8, Actor and corresponding capabilities

3.3.4 Operational Object

Rationale

In order for people to flawlessly collaborate, it is essential to have an unambiguous understanding of the things we use and communicate about. These are to be captured by strict definitions as operational objects. Since the objective of the description of the operational aspect is to understand the content, structure and

behaviour of the Mission Space, these objects provide definitions of ‘things’ in the real world, and are the starting point for the informational description we seek to design in automated systems.

Definition

An operational object is a thing or entity that occurs in the real world and of which information is required about the facts that need to be know. It can be a tangible object, like military equipment, an event such as an operation or a concept like a directive.

Objective

The objective of capturing ‘objects’ relevant to an organisation is to establish unambiguous definitions for them. Having definitions allows the different collaborating parties to refer to the exact same thing, thus enabling clear communication.

Description

The operational objects are captured as a formal specification of the entities that can exist for a considered system. By arranging the relationships between the various operational objects consistent and unambiguous terminology is established. The formal specification consists of a description and definition of the considered operational objects and the coherence of those definitions.

Capturing the operational objects is essential to arrive at a correct understanding of the operational aspect of an architecture, thus essential for creating the correct, unambiguous and complete communication of an organisation.

The existence of operational objects can be dependent on other objects. E.g. assuming that the operational object ‘Trip’ represents the travelling by car according to a certain route, the operational object ‘Trip’ is dependent on the operational objects ‘Car’ and ‘Route’. In order to define the operational objects ‘Trip’, both operational objects of which it depends need to be defined as well. If the operational object ‘Car’ is not relevant to the considered system, the operational objects ‘Trip’ cannot be defined. The universe of discourse determines the scope of the operational objects that need to be defined. Things that are outside the environment of a system do not need to be defined.

Capturing the operational objects is essential to arrive at a correct understanding of the Mission Space, thus essential for creating the correct, unambiguous and complete communication of the considered system.

Added value

The added value of capturing objects and corresponding information objects is that it allows:

- To delimit the scope of a domain by identifying all the operational objects in the domain.
- To establish unambiguous definitions of operational objects, enabling a common and better understanding and a common unambiguous language in the domain.
- To create an ideal start for the identification and description of information and data necessary to perform the collaborations in the considered system.
- To describe the universe of discourse in terms better understood by the people operating for the organisation.
- To help assign responsibilities for the quality of shared data.

Similar terms

For the architecture element ‘Operational object’ the following similar terms are used:

- Entity
- Fact
- Business object
- Business entity
- Operational entity
- Operational object

Covered aspects

Since an operational object is described in order to have a better understanding of the mission space, it is part of the operational aspect.

Corresponding architecture description level

An operational object is an implementation independent architecture element describing what things are. Therefore this element is part of the functional description level.

Example

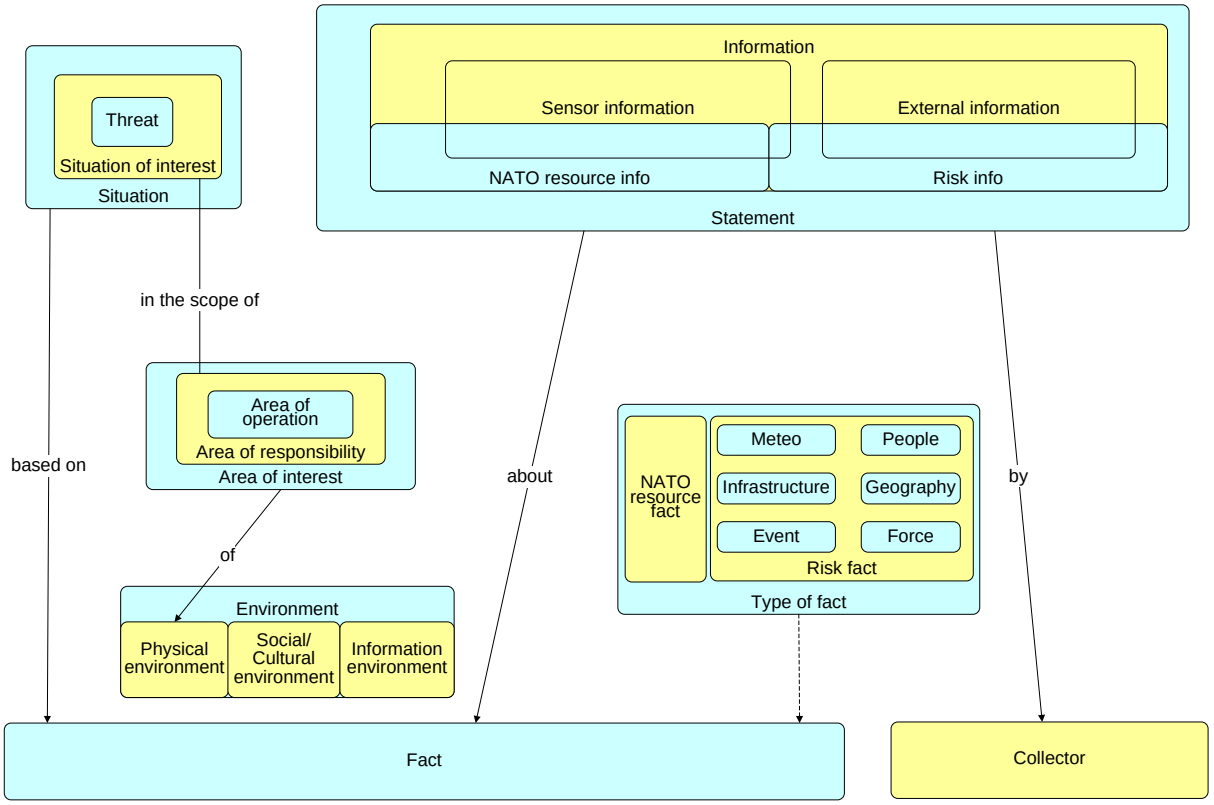

Figure 3-9, Operational objects from Overarching Architecture v2.5

3.3.5 Information Object

Rationale

Information objects represent the facts that need to be known about operational objects and their coherence, in order to turn the set of representations into information. Information objects are the information image of events, concepts and tangible things. The context for this information image is directly related to the operational objects described in the mission space. Normally, not each and every detail is relevant to the considered system. Determining which properties of the operational objects are relevant and need to be captured as data, is essential. The considerations are guided by the context as captured in the mission space architecture descriptions.

Definition

An information object is an implementation independent representation of the facts that need to be known about objects and their coherence in order to turn the set or representation into information.

Objective

The objective of capturing information objects is to identify and unambiguously describe all the elements of information and their properties that are relevant for execution of tasks in the mission space.

Description

Capturing the information objects is essential to arrive at a correct understanding of the information aspect of an architecture, thus essential for creating the correct, unambiguous and complete communication of an organisation.

An information object provides a formal specification of the facts that are relevant to know about objects. These specifications represent the infological relationships between information objects, whereas the definitions of objects refer to semantic relationships between objects.

Added value

The added value of capturing objects and corresponding information objects is that it allows:

- To provide a complete set of information definitions.
- To provide a common and better understanding between parties by creating an unambiguous language.
- To serve as a foundation for implementation focused data models.
- To enable linking existing data models like the NATO Corporate Data Model (NCDM) and the Joint C3 Information Exchange Data Model (JC3IEDM).
- To support communication with subject-matter experts.

Similar terms

Similar terms for the term ‘Information object’ are:

- Entity
- Information entity
- Information element

Covered aspects

Since an information object is captured to understand the information aspect, it covers the information aspect.

Corresponding architecture description level

An information object is an implementation independent architecture element describing what the specification is of the facts that are relevant to know about operational objects. Therefore this element is part of the functional description level.

Example[2]

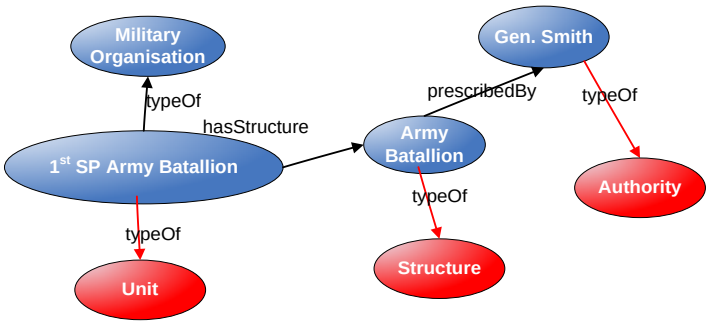

Figure 3-10, Information Objects

3.3.6 Process

Rationale

For the delivery of the required results and effects, there needs to be a correct supporting pattern of tasks and actions, conducted by the right types of person, with the right qualifications and abilities, using the right capabilities. The collaboration between the types of persons, also called actors, follows a certain time-bound sequence. This is usually referred to as a business or an operational process.

In view of the irreversible impact of the processes in a military environment, there is little room for failure. Bases on process descriptions, specific processes can and need to be tailored for specific missions and operations.

[2] From: Semantic Interoperability, Victor Rodriguez-Herola, NC3A, 7 December 2006

Definition

A process is composition of activities that are triggered by an event and transforms a specific input into a meaningful output.

Objective

The objective of capturing ‘processes’ is to understand how activities are orchestrated in order to be able to deliver the required results and effects by describing the coherence and sequences of those activities that are performed by the actors.

Description

A process ties together the patterns of collaboration and the coherency between activities. Furthermore, a process defines the sequences of the coordination activities (C2) and effects creating activities. The process description does not refer to actual implementations. It describes the process in generic terms that can and need to be tailored to a specific mission.

Added value

The added value of capturing processes is that it allows:

- To assign tasks and responsibilities to individuals.
- To qualitatively assess the abilities and skills required by the required resources.
- To validate the execution of a type of mission.
- To tailor a generic well-defined solution to a specific situation, thus providing a fast mechanism for creating sound processes.
- To consider the functionality of automated systems’ support against its operational use and users.

Similar terms

For the architecture element ‘Process’ the following similar terms are used:

- Business process
- Operational process
- Operational activity
- (Business) Function

Covered aspects

Since a process describes how effects and results are delivered, a process is part of the operational aspect of an architecture.

Corresponding architecture description level

A process describes how the organisation pursues its goals and objectives. In other words, processes indicate how actors need to perform in order to deliver the required set of effects. Therefore, the architecture element ‘process’ is part of the construction level.

Example

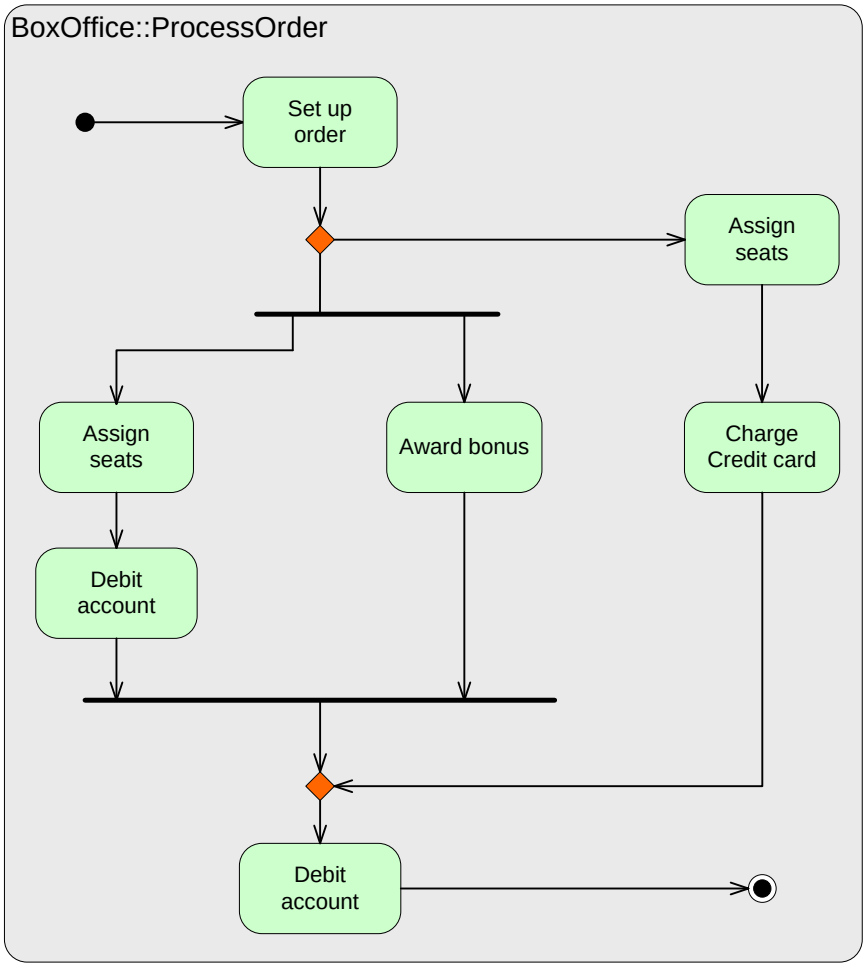

Figure 3-11, Processes (UML Activity diagram)

3.3.7 Location

Rationale

In a NATO environment missions and operations are conducted from several locations. Actors can perform their tasks and execute their processes on various types of locations. Since actors are implementation independent, it is important to identify what possible locations where the actors perform their tasks.

Definition

A location is a geographical spot, e.g. a place represented by spatial coordinates.

Objective

The objective of capturing ‘locations’ is to understand where activities are executed by the actors. The same actor can carry out the same on different locations.

Description

Locations can be categorised using different types of location. The classification can be defined based on unity of task execution, e.g. on the basis of properties relevant to the execution of a mission. Examples of type of locations are:

- Static type of location: fixes to the ground in one location over a longer period of time, e.g. NATO Headquarters (HQ).
- Deployed static type of location: operate in theatre; require stationary for operations, e.g. International Security Assistance Force (ISAF) HQ in Afghanistan.
- Deployed type of location: operate in theatre; operate while moving (can also operate be temporary stationary), e.g. aircraft carrier.

Added value

The added value of capturing locations is that it allows:

- To identify which combination of actors conducting their tasks on what locations.
- To act as a solid starting point for allocation of staff and resources.
- To serve as a stable starting point for organisation design.
- To serve as a starting point for the implementation of information flows.
- To illustrate similarities in information flows actors on different locations.

Similar terms

For the architecture element ‘Location’ the following similar terms are used:

- Location type
- Node

Covered aspects

A ‘location’ describes an operational issue of the architecture and therefore is part of the operational aspect.

Corresponding architecture description level

Since ‘location’ describes where actors carry out their tasks and processes, this architecture element comprises of the construction level.

Example[3]

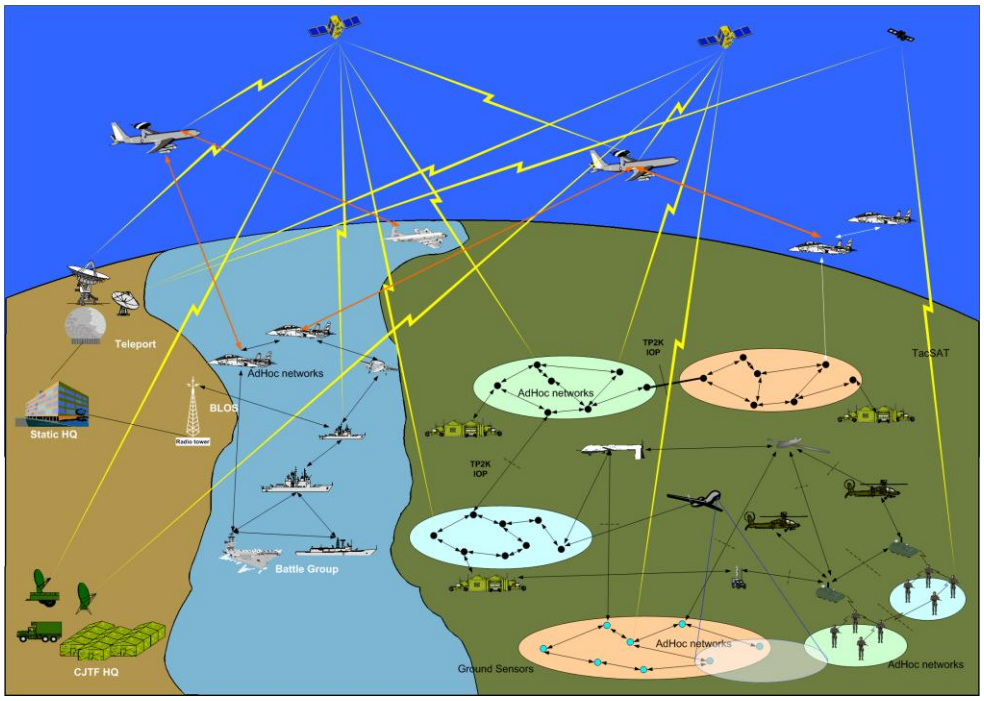

Figure 3-12, Locations for an NNEC environment

[3] From: Semantic Interoperability, Victor Rodriguez-Herola, NC3A, 7 December 2006

3.3.8 Information Requirement

Rationale

Information needs to be presented to humans such that they can understand it. This requires the combination, collation and exchange of elementary pieces of data into a consumable package. This package can take any form: paper, electronic, graphical, text, voice, video, interactive 3D map, etc. In fact it is not the physical implementation that is most interesting, but rather understanding the required composition of the pieces of data and the quality constraints that come with it.

Definition

An information requirement is a statement of what an actor needs to know about objects in order to fulfil his tasks in the mission space.

Objective

The objective of capturing ‘information requirements’ is to identify the information required, and the data produced as relevant to Actors. Furthermore, information requirements are captured to understand the requirements for the information exchange between actors.

Description

Information requirements state the basic ‘information functionality’. In other words, what does each actor need to know to achieve his objectives and deliver his services of the organisation? The information requirements description consists of the relevant pieces of information that provide certain insights to the actors inside and outside the organisation, including the basic facts these entities are built upon: information elements and their constituting information objects. All the descriptions are produced taking into account the perspective of the actor using the information. Therefore, the starting point of the description is the usage of that information by an actor.

Added value

The added value of capturing requirements is that it allows:

- To provide the complete set of information needs per actor.
- To provide a starting point for identifying characteristics of information objects.
- To serve as a foundation for developing new and linking existing data models.
- To support communication with stakeholders and subject-matter experts.

Similar terms

For the architecture element 'information requirement' the following similar terms are used:

- Requirement
- Needline
- Information exchange requirement

Covered aspects

An 'information requirement' describes what information needs to be exchange. It is therefore obvious that this element covers the information aspect of an architecture.

Corresponding architecture description level

Since an 'information requirement' refers to what information is needed, it is part of the functionality description level.

Example[4]

Information Exchange Requirement (IER)	From Activity	From Actor	To Activity	To Actor
Data Acquisition Request	Nominate	Control	Manage	Observer Operator
Requested Target Data	Manage	UAV Operator	Manage	Target Contract
Requested target data	Manage	UAV GPS Technician	Manage	Target Contract
Requested target data set	Manage	Target Contract	Issue fire	Control
Target data	Manage	UAV Service provider	Manage	UAV Operator
Target data	Manage	GPS	Manage	UAV Operator
Target data	Manage	GPS	Manage	UAV GPS Technician
Target data retrieval request	Nominate	Control	Manage	Target Contract
UAV position	Nominate	Control	Manage	UAV Operator
UAV position	Nominate	Control	Manage	UAV GPS Technician
UAV position data	Manage	UAV Operator	Manage	UAV Service provider
UAV position data	Manage	UAV Operator	Manage	GPS
UAV position data	Manage	UAV GPS Technician	Manage	GPS
UAV schedule entry	Manage	Observer Operator	Manage	UAV Service provider
UAV schedule entry	Manage	Observer Operator	Manage	GPS

Figure 3-13, Example Information exchange requirements

[4] From DoD Framework overview, EA Frameworks, LLC 2004

3.3.9 Business Rule

Rationale

In order to be effective every organisation must comply with certain rules. Those business rules indicate what kind of behaviour of the organisation is required for the realisation of the desired effects. Business rules are the translation of the operational strategy, legislation and policies to operational guidelines and directives. Those rules are not only applicable for the processes and supporting information systems.

Definition

A business rule is a constraint that determines the behaviour of an enterprise while achieving its goals and objectives.

Objective

The objective of capturing the business rules is to identify and understand what constraints are for processes, operations and the supporting information systems in order to achieve the required effects.

Description[5]

A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behaviour of the business. The business rules which concern the project are atomic ~ that is, they cannot be broken down further. This may be viewed from two perspectives. From the operational perspective, it pertains to any of the constraints that apply to the behaviour of people in the enterprise. From the information system perspective, it pertains to the facts which are recorded as data and constraints on changes to the values of those facts. That is, the concern is what data may or may not be recorded in the information system.

Business rules exist for an organisation whether or not they are ever written down, talked about or even part of the organisation's consciousness. However for an organisation that transforms towards an effect-based organisation, it is advisable to gather business rules in formal manner. Organisations may choose to proactively describe their business practices in a database of rules. More commonly, business rules are discovered as part of a formal requirement gathering process. In this case the collecting of the business rules are coincidental.

[5] Based on: Defining Business Rules-What Are They Really?, The Business Rule Group, Final Report July 2000 and Wikipedia-Business Rule

Capturing and implementing the business rules increase the flexibility and agility of an organisation. In case of changes in policy or legalisation an organisation that has formalised its constraints, can adjust much faster and more effective to the new situation. Also the impact on information systems is far more traceable.

Added value

The added value of capturing business rules is that it allows:

- To help to understand operational strategies and policies.
- To help identifying operational and information requirements.
- To serve as a sound foundation for identifying and defining quality factors for both operations (quality of operations) and information systems (quality of information and quality of service).
- To serve as a starting pointing for defining the knowledge of the organisation.

Similar terms

For the architecture element ‘business rule’ the following similar terms are used:

- Rule
- Entity type rule
- Condition

Covered aspects

A business rule can have two perspectives: the business perspective and information system perspective. The business rules for the business perspective are part of the operational aspect, whereas the business rules for the information system perspective are part of the information aspect.

Corresponding architecture description level

Business rules describe what the constraints are for the behaviour of the organisation. Therefore this architecture element is part of the functional description level.

Example

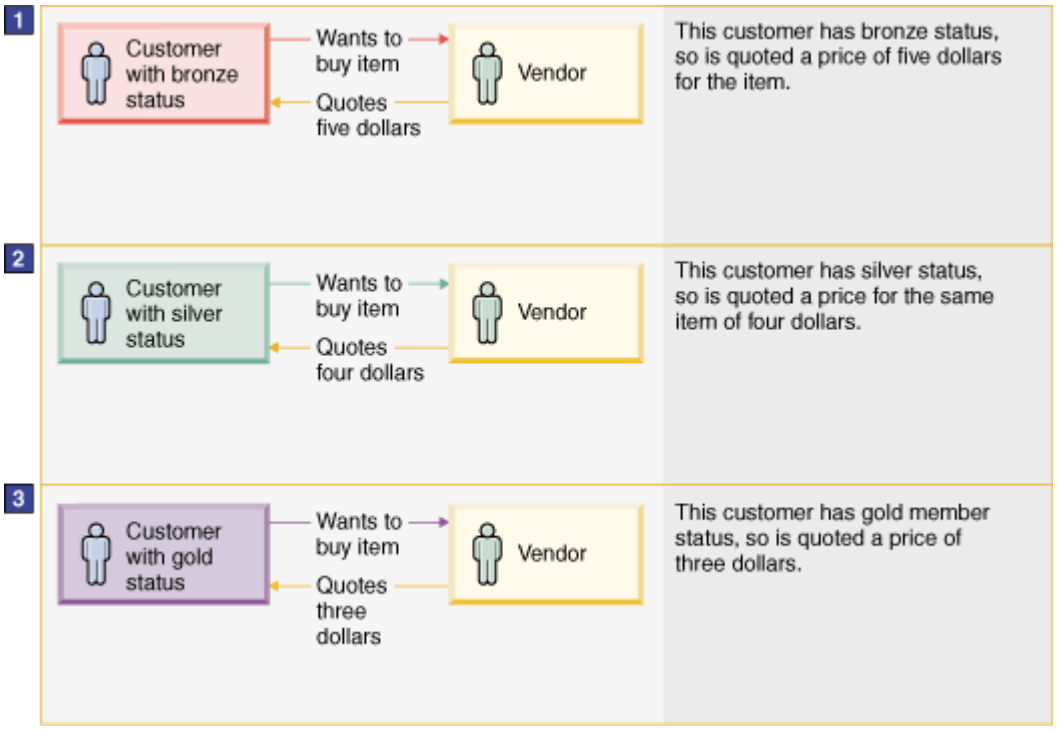

Figure 3-14, Business Rule diagram[6]

3.3.10 Information Product

Rationale

Information needs to be provided to the actor that uses it while performing his role. This is conducted by delivery of information products. Information products represent the right information in the right format, based on processing of information objects. The processing of information objects can be calculation, collation, combination, etc. Information services also represent the distribution of information from the storage to the actor.

Definition

An information product is a meaningful output comprising an implementation independent representation of required information for the achievement of goals and objectives.

[6] From IBM, Websphere Integration Developer

Objective

The objective of capturing information products is to gain an understanding of the information required by actors and how that information needs to be delivered.

Description

People in the organisation need the right information delivered on time and in the right format. The combination of required information at the right time and in the right format is called an information product. What information products are needed and when these are needed depends on the processes that are executed by the actors. Therefore it is relevant to determine at what stage of the process execution information products are required.

Each information product needs to be able to meet the Quality of Information (QoI) constraints that are linked to the information objects. Additionally, QoI constraints need to take into account the potential distribution of people in the organisation.

Added value

The added value of capturing information product is that it allows:

- To have a comprehensive view of the information distribution to actors.
- To prescribe and validate the quality constraints those need to be met in considering end-to-end information sharing.
- To support a sound decision on what functionalities can be automated.

Similar terms

For the architecture element ‘information product’ sometimes the term ‘product’ is used.

Covered aspects

An ‘information product’ describes, as the term indicated, what information is needed in what format. It is obvious that this architecture element is part of the information aspect.

Corresponding architecture description level

An information product covers when and in what format the required information needs to be delivered; it describes how the information is represented. Therefore this element is part of the construction description level.

Example

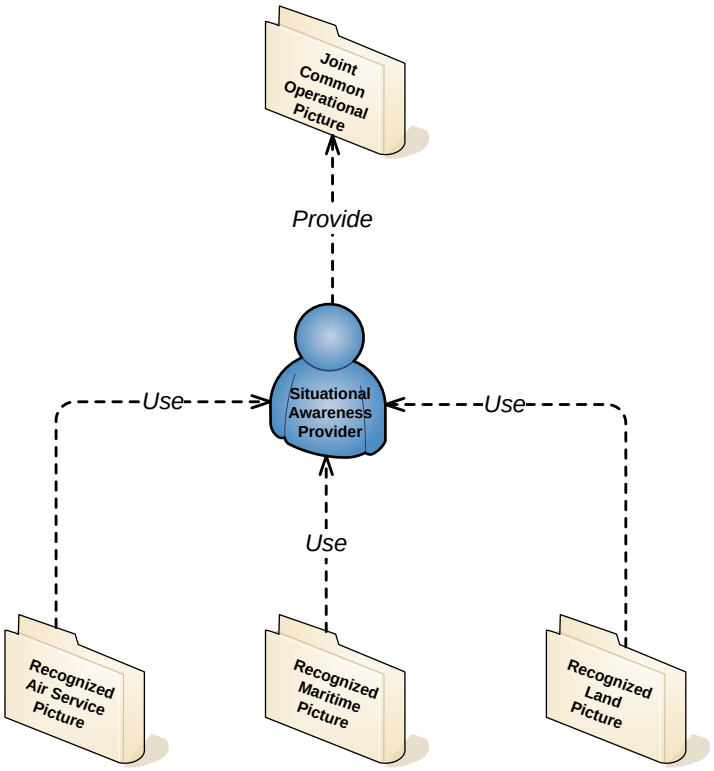

Figure 3-15, Elements of Joint Common Operational Picture

3.3.11 Information Flow

Rationale

The end-to-end support for the creation of data and provision of information, determines important facets of the complete functionality of the automated information system and its technical infrastructure. This end-to-end link does not need to be implemented as a one-to-one relationship. In fact in a service-oriented paradigm, that is typically not the case. However, it is expected that due to specific Quality of Information constraints, current technology requires a direct link between the users of information. By identifying the information flows, a sound decision can be made on this issue.

Definition

An information flow is the full chain of end-to-end information delivery from capturing, processing, and storing of data by one set of actors to processing, distributing and using information by another set of actors.

Objective

The objective of capturing information flows is to establish the construction of the communication patterns between groups of actors by decoupling data production from information usage. Secondly, identifying the information flows enables the identification of Quality of Information constraints for the information delivery.

Description

Information flows describe the patterns of communication between (groups of) actors as the chains of information delivery invocations needed in order to produce the required information. Information flows also describe the Quality of Information constraints for the information needed by the actors in order to fulfil their tasks. The Quality of Information for the full pattern must be consistent, where the information using actors dictate the strictest constraints that need to be met by the set of providing information deliveries.

Added value

The added value of capturing information flows is that it allows:

- To verify if the actors' needs for information can be met in compliance with the Quality of Information constraints.
- To understand the end-to-end information creation and use, while allowing a completely decoupled implementation.
- To support the decision-making on which end-to-end chains need to be implemented as one-to-one links, while others can be implemented as decoupled information deliveries.

Similar terms

For the architecture element 'information flow' the term 'information exchange' is also often used.

Covered aspects

An 'information flow' describes, as the name ready suggests, a part of the information aspect.

Corresponding architecture description level

An information flow describes the way actors deliver and receive information and information products. In that sense it describes how the information exchange is constructed. This element is part of the construction description level.

Example

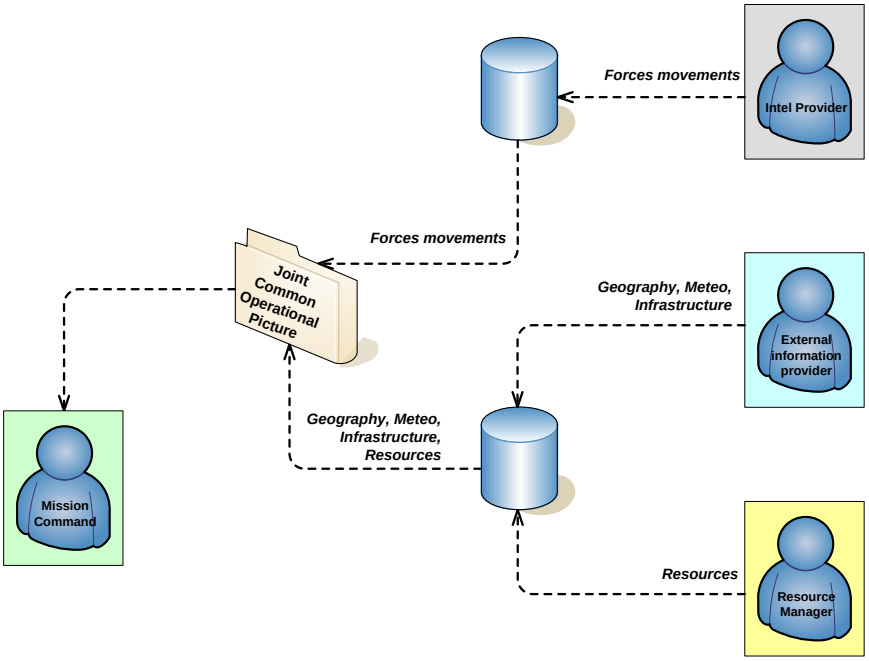

Figure 3-16, Example information flow

3.3.12 User

Rationale

A user can be seen as an individual or group of individuals requiring the same set of automated functionality. In that sense the term user, that is related to automated applications) needs to be distinguished from the term actor, that is related to tasks and responsibilities. This distinction creates flexibility in designing architectures and systems; new or other automated functionalities can be allocated to the same actor, while fulfilling its responsibilities.

Definition

A user is a person or a group of persons representing one or more actors using the same functionality of an automated application.

Objective

The objective is to identify the tasks related to an Actor that need to be supported by automated functionality.

Description

As an architectural design decision, tasks performed by actors can be turned into fully automated and self-supporting pieces of software that operate without continuous human intervention. This can typically be the case where the Quality of Information constraints are such that it is impossible for a human to meet the requirements. However, in all cases a person will remain responsible for the execution of the automated solution and therefore needs to remain in control of the service.

Users represent those actors that actually use automated applications. Not for the fulfilment of every responsibility an actor needs to be supported by an application. The term ‘user’ indicates that the corresponding actor is using a part of the application. An actor can be represented by multiple users. The opposite case is also valid; a user can represent multiple actors.

It is crucial to realise that the term ‘user’ is defined form the point of view of the automated application.

Added value

The added value of capturing users is that it allows:

- A comprehensive view of the tasks of each actor that are to be supported by automated functionalities.
- A clear decision on what is to be automated and what isn’t.

Similar terms

For the architecture element ‘user’ sometimes the terms ‘consumer’ is used.

Covered aspects

As earlier stated the term ‘user’ is defined form the point of view of an automated application. Therefore, user is a concept covering the technology aspect.

Corresponding architecture description level

A user is an implementation independent concept, depicting what the responsibility towards automated tasks is. Therefore the architecture element ‘user’ consists of the functional level.

Example

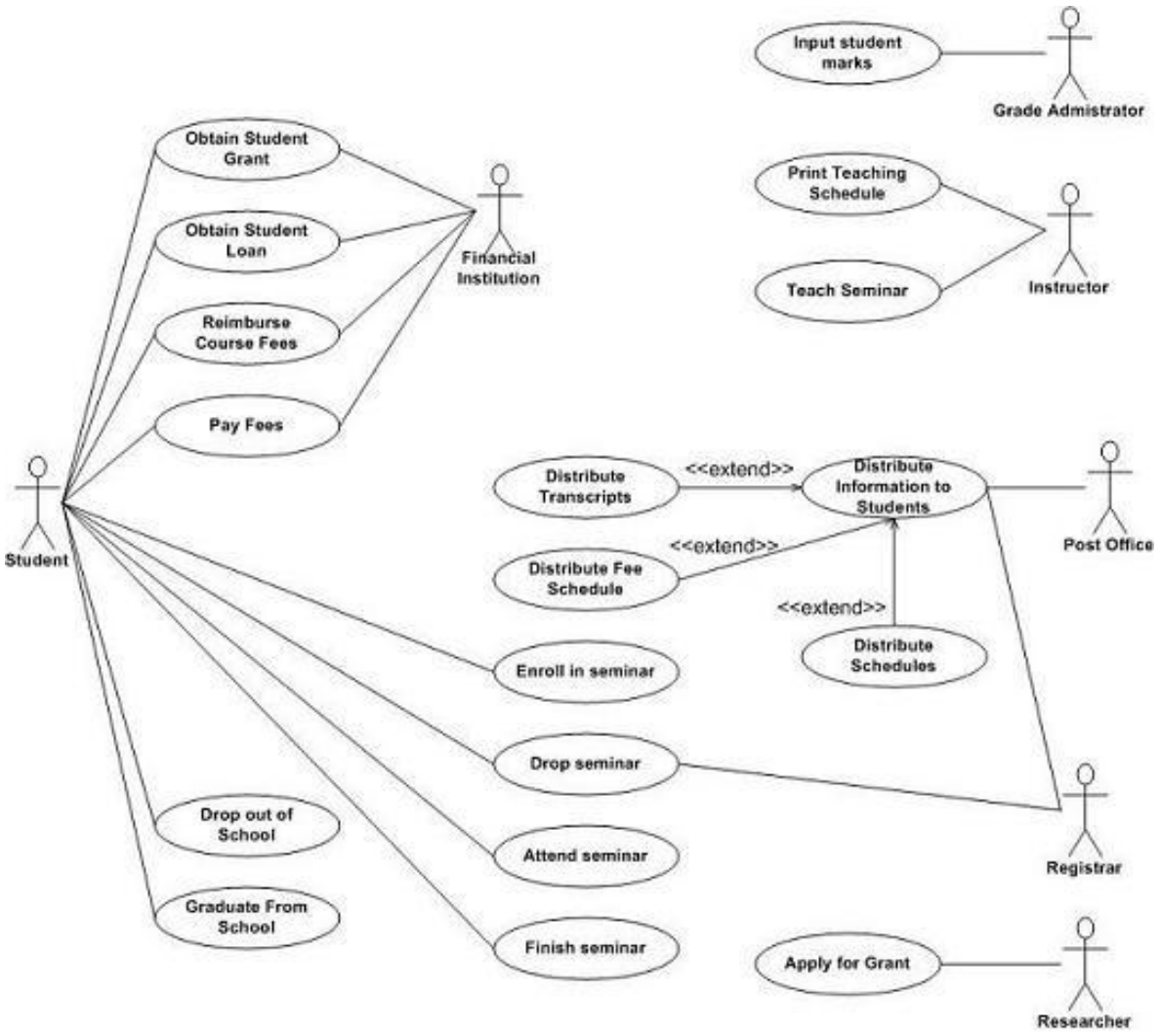

Figure 3-17, Users represented in UML’s Use case Diagram

3.3.13 Service

Rationale

As already mentioned in the description of the NNEC Architecture Concepts (refer to 3.2.1) service-orientation is an important paradigm for the NNEC transformation. Obviously, it makes sense to identify the concept ‘service’ in an NNEC transformation environment. Since the descriptions of NNEC architecture elements are reflected in three architecture aspects, namely the operational, the information and the technology aspect, three corresponding concepts of services are identified: an operational service, an information service and an application service.

Definition

A service is implementation independent specification of the deliverables that has added value to the consumer of that service in accordance with the consumer’s goals and objectives.

An operational service is a service providing an observable outcome which fulfils an operational need.

An information service is a service providing data which fulfils information requirements.

An application service is a service delivering automated functionality which fulfils the needs and requirements of the user, provided by an automated application.

Objective

The objective of capturing the operational services is to gain a thorough understanding of the concept of operations as a basis for identifying the collaborations in the organisation. This understanding is independent of the way in which the operational services are implemented, thus establishing a single point of reference for the operational activities, while allowing room and flexibility for different implementations.

The objective of an Information Service is to capture the delivery of information to actors.

The objective to identify the application services is to support the tasks of the user. Each application service represents an independent and delineated piece of functionality. The identification of application services includes the identification of domain logic, like screening rules for inputting data, and algorithms for the production of (new or updated) data.

Description

An operational service must lead up to an observable and measurable outcome. The execution of an operational service needs to comply with quality of operation requirements. Typical requirements are speed of execution, and adaptability to changing situations.

An information service describes the delivery of information to an actor. For each operational service and corresponding actor, an information service describes what information is required and used in a process, and which Quality of Information constraints need to be met.

An application service describes the comprehensive set of functionalities provided in an automated fashion from the point of view of the user for a distinct task. For each user, there will be a collection of application services based on the description of the users as described in section 3.3.12.

Added value

The added value of capturing operational, information and application services is that it allows:

- To delimit the scope of the operational contents of the mission space.
- To understand the service portfolio of the considered system.
- To have a better understanding of the interdependencies between parties.
- To act as an entry-point to identify needed capabilities.
- To create a basis for agreements with the Nations and other contributors in order to enable the sourcing discussion ‘what does NATO need to do by itself and what could/should other parties do’.
- To help assign tasks to actual persons and organisational entities.
- To serve as a stable starting point for process and organisation design.
- To have a comprehensive view of the information distribution to persons.
- To prescribe and validate the quality constraints those need to be met in considering end-to-end information sharing.
- A comprehensive view of the automated functionality each user requires.
- The possibility to analyze and optimize the application services that are required and establish a catalogue of application services.

Similar terms

For the architecture element ‘services’ the following similar terms are used:

Pertaining to operational service:

- operational function
- business service

Pertaining to information service:

- information function
- functional service

Pertaining to application service:

- user service
- information technology service
- technical infrastructure service
- NNEC service

Covered aspects

The term ‘service’ is related to all three architecture aspect of which the operational service covers the operational aspect, the information service covers the information aspect and finally the application service covers the technology aspect.

Corresponding architecture description level

A ‘service’ is an implementation independent specification of the delivery of a provider to a consumer. Therefore, the concept ‘service’ is part of functional description level.

Example

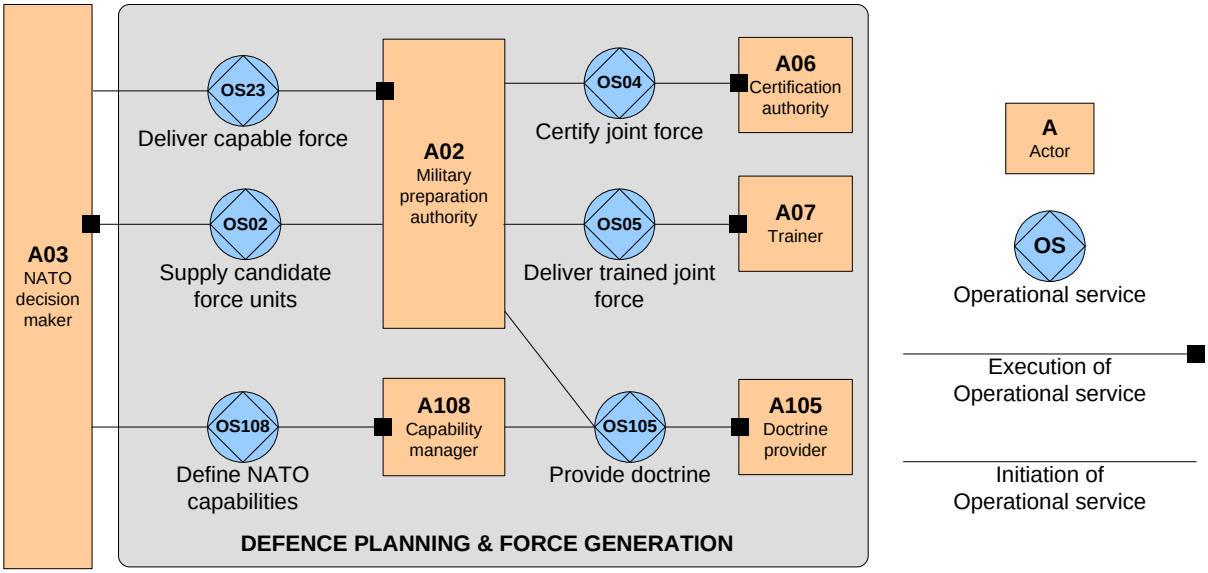

Figure 3-18, Example operational services

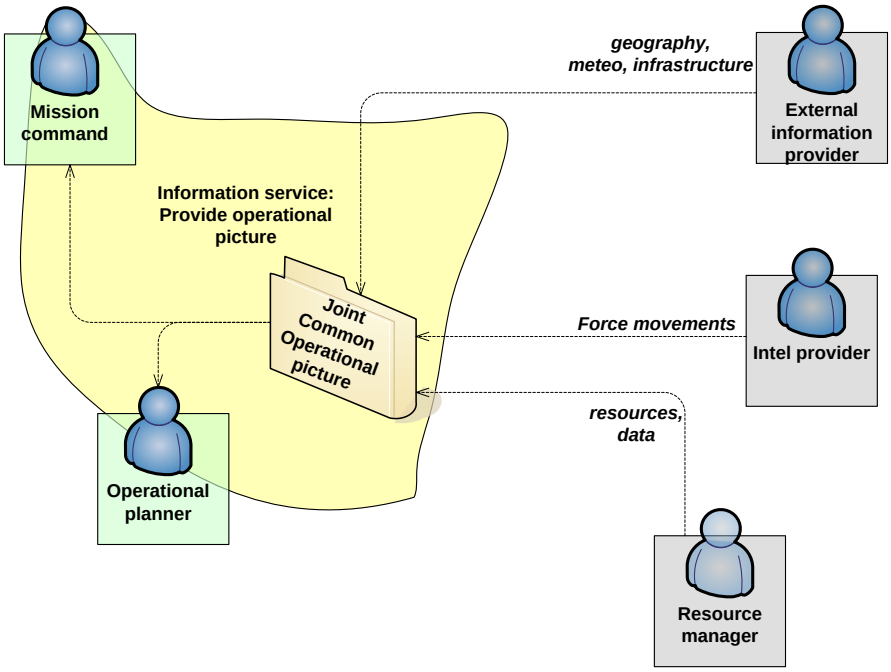

Figure 3-19, Example information service

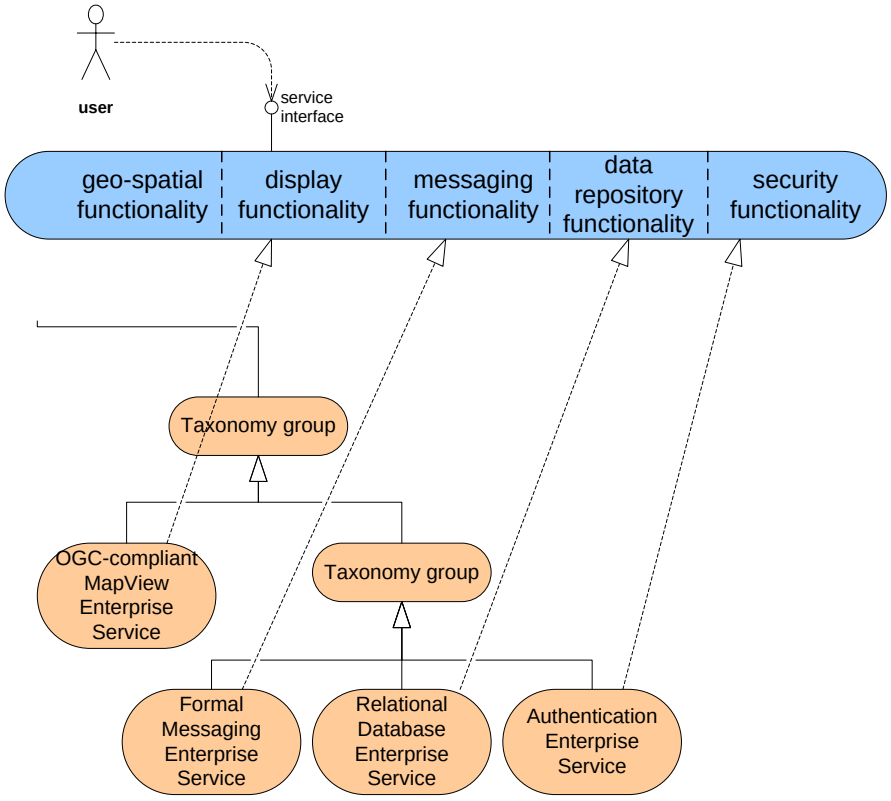

Figure 3-20, Example of a geo-type application service, related to a service taxonomy

3.3.14 Component

Rationale

Application services are constructed by different types of components, e.g. user interface components, data access components, network components, servers, satellites. Two types of components can be distinguished: application components and technical infrastructure components. The combination of these application components that is selected to fulfil the required functionality is dependent on several factors. The Quality of Service (QoS) constraints for the application services play an important role in selecting the right combination of components. Another consideration deals with variability of implementation. If the application service is to be used at different types of locations, this might lead to different implementations, e.g. the construction of the solution for use in a mobile situation will be different form the construction of the same application service in a static environment. The variability of implementation needs be hidden as much possible to other components.

Definition

A component is the logical construction element of an elementary application service. An application component is a software component, whereas a technical infrastructure component is a hardware component.

Objective

The objective of a component is to specify the internal behaviour and construction of each application service in order to fulfil the required functionality, taking into account the conditions and the types of resources that govern the application of the components. Secondly, this model captures the Quality of Service (QoS) requirements the User Service needs to adhere to.

Description

A component describes the content and the distribution of responsibilities of application and technical infrastructure components as derived from an elementary application service.

A component encapsulates the variability in implementation. Thus the logical and technical construction of an application service will not vary, but the implementation of a component can.

A single component can and preferably will be used in the construction of different application services.

Components have to adhere to the Quality of Service (QoS) requirements for the application services in order to meet the functionality requirements of the user. If an component is to be used in multiple application services, the strictest QoS requirements prevail. The Quality of Service (QoS) requirements can be determined from the Quality of Information (QoI) of the information flows by reasoning backwards from the Quality of Information (QoI) related to the delivery of the information, establishing the Quality of Service (QoS) required reaching that Quality of Information (QoI).

In principle, there are 3 types of application components that can be used to construct an elementary application service. This is based on the common 3-tier architecture pattern:

- User interaction components: handle the interaction (input, output) with the user.
- Business Logic Components: handle the implementation of a business rule.
- Persistency Components: handle the storage and retrieval of data.

Furthermore there are also 3 types of technical infrastructure components. This is based on the common client-server network topology:

- Devices: infrastructure components that are intended for presenting and entering information, such as computers, satellites, handheld devices.
- Network edge components: infrastructure components, such as routers, transmission equipment, and edge servers, such as proxy servers, firewalls and load balancers.
- Internal network components: infrastructure components to construct internal networks, such switches or to run and maintain applications, such as web servers, application servers, middleware servers and database servers.

Added value

The added value of capturing components is that it allows:

- To cater for implementation variability while keeping the logical construction stable.
- Substitution of different implementation products.
- Identification of the application of off-the-shelf hard- and software.
- To provide a clear linkage to (candidate) data base systems.
- A sound starting point for systems and technical design.
- To provide a comprehensive view of dependencies of technical facilities.
- To identify the quality of the technical components.

Similar terms

For the architecture element ‘components’ the following similar terms are used:

Pertaining to application component:

- web service
- software component
- user application

Pertaining to technical infrastructure component:

- hardware component
- user device
- systems hardware

Covered aspects

Since a 'component' describes how hard- and software is used for the construction of an application service, it is part of the technology aspect of an architecture description.

Corresponding architecture description level

Due to the fact that a component describes the logical and technical construction of an application service, it is part of the construction description level.

Example

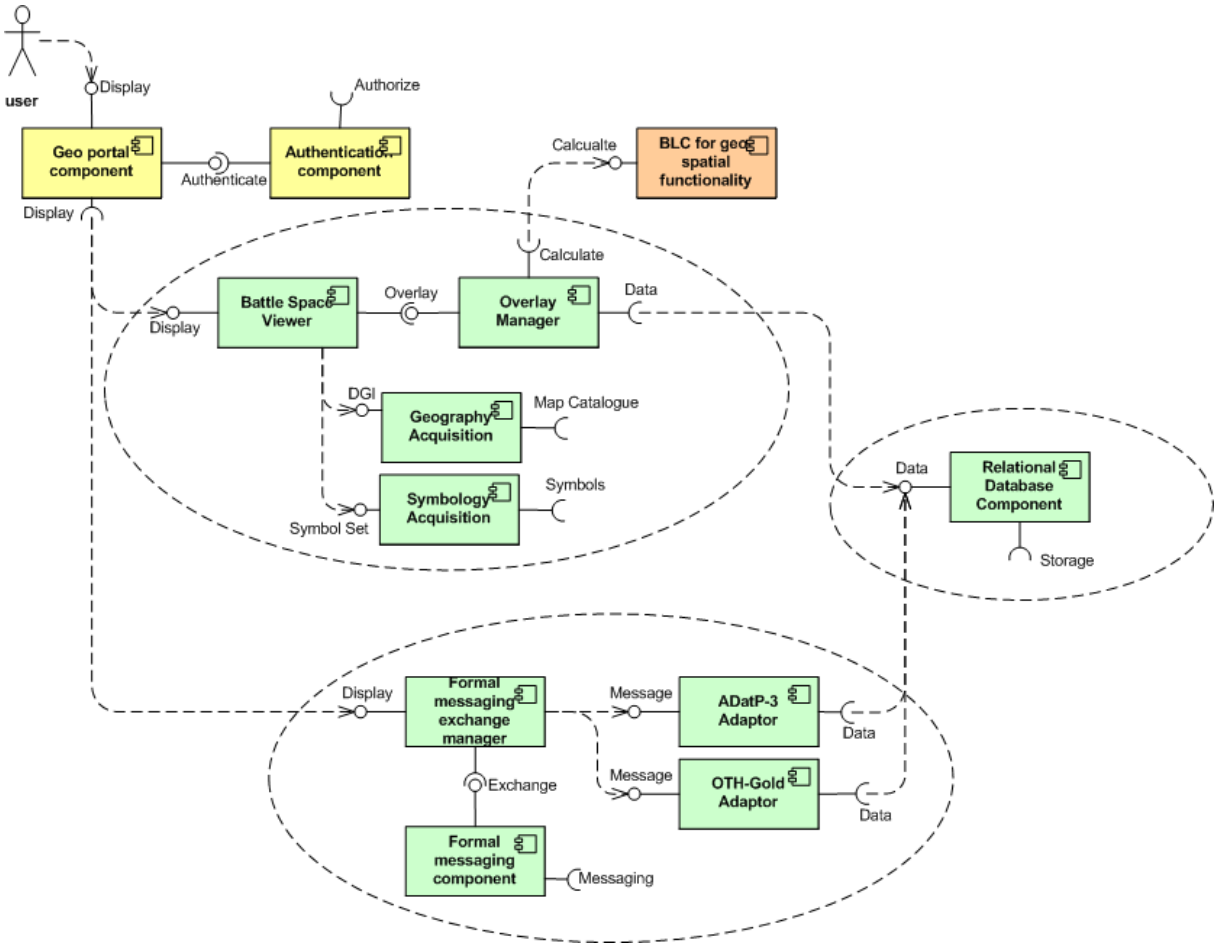

Figure 3-21, Example components (here shown as a set of collaborating components)

3.3.15 Component Collaboration

Rationale

Similar to the considerations for selecting the right components are the considerations for the collaboration between components. In order to arrive at a sound construction, some specific collaboration between components can be essential, e.g. in order to execute a specialized algorithm the relevant data needs to be retrieved first. Furthermore, it is important to identify the impact of components over the different location types on the adequate functioning of an application service of the distribution. While identifying these collaborations, a need for additional components may occur, e.g. components that provide distribution of data.

Definition

A component collaboration is the necessary cooperation of two or more components in order to provide a required functionality.

Objective

The objective of capturing component collaborations is to specify how the collaboration between components needs to take place in order to fulfil the functionality defined in the application services.

Description

Component collaborations capture dependencies between components, indicating how the construction of an application service is intended to work.

There are two essential aspects that need to be captured for a component collaboration:

- Invocation: the (logical) way(s) to activate the component. This includes the data the component needs to process on behalf of the invoking entity (parameters). However, the manipulation (retrieval, processing and storage) of local data is encapsulated in the component and therefore hidden to the invoking entity.
- Mechanism: the nature of the collaboration between this component and the invoking entity. The types of applied information exchange technology (inter-process communication) and communication technology are defined here.

Since there are two types of components, two types of component collaborations are identified as well: application component collaboration and technical infrastructure component collaboration.

Added value

The added value of capturing component collaborations is that it allows:

- To provide a comprehensive view of the dependencies between the components.
- To provide a comprehensive view of shared construction elements and technical facilities.
- To provide an implementation independent specification of components and interfaces.
- To provide traceability of each component through the full development life-cycle.
- A sound starting point for systems and technical design.

Similar terms

For the architecture element ‘component collaboration’ the following similar terms are used:

- component interaction
- interfacing
- component data exchange

Covered aspects

Like components, component collaboration describes how hard- and software is used for the construction of an application service and there is part of the technology aspect of an architecture description.

Corresponding architecture description level

Due to the fact that a component collaboration describes the logical and technical construction of an application service, it is part of the construction description level.

Example

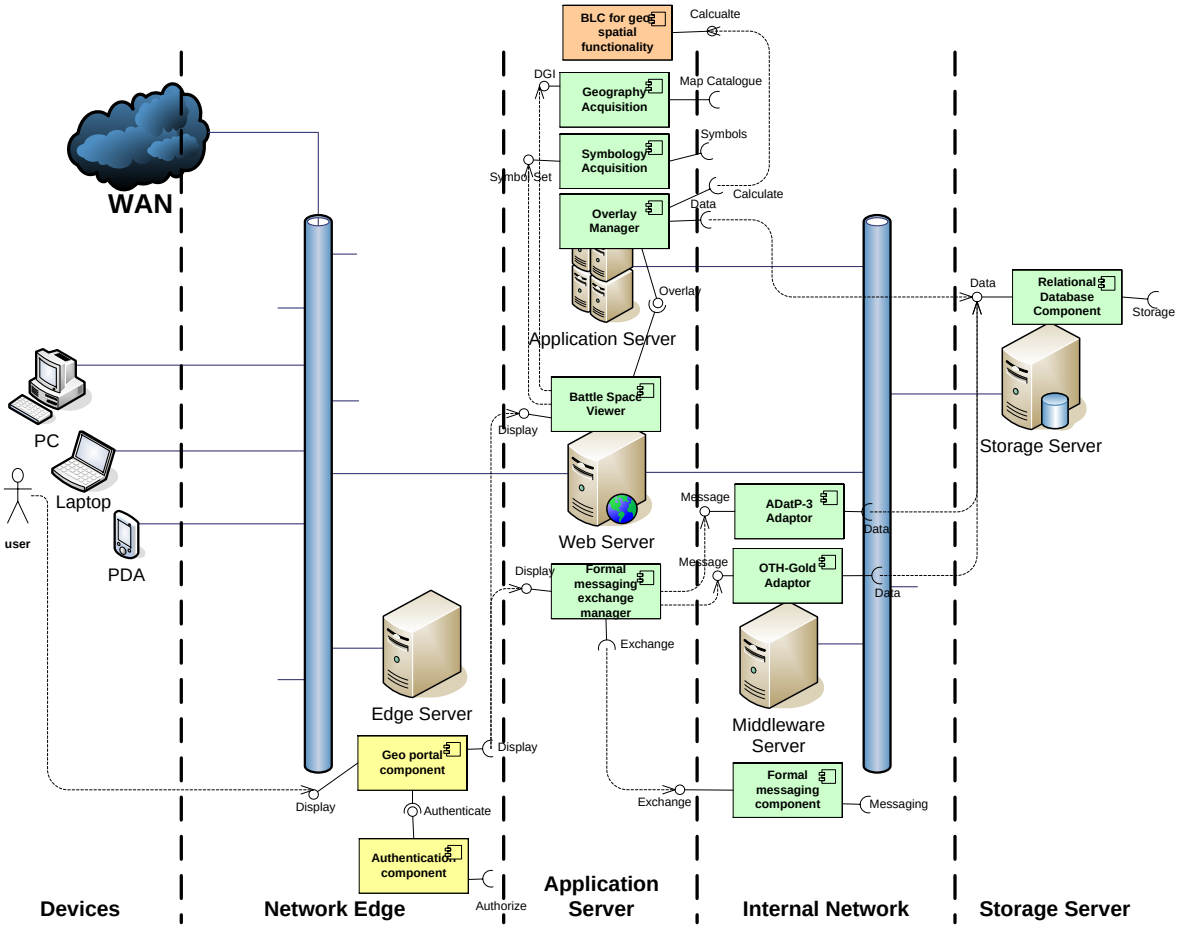

Figure 3-22, Example component collaboration (here shown as a component collaboration with components distributed across a typical local area network)

List of Figures

This page is left blank intentionally.